НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

(повне найменування інституту, факультету)

Автоматизованих систем обробки інформації і управління

(повна назва кафедри)

«До захисту допущено»
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис) (ініціали, прізвище)
“ ” 2020 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Програмне забезпечення інформаційних управляючих систем та технологій»

спеціальності «121 Інженерія програмного забезпечення»

на тему *Веб-застосування для виявлення аномалій в електрокардіограмах лінгвістичним меодом*

Виконав: студент IV курсу, групи *ІП-61 Цицилюк Анна Валеріївна* _____
(прізвище, ім’я, по батькові) (підпис)

Керівник *ст.викладач Олійник Юрій Олександрович* _____
посада,науковий ступінь,вчене звання,прізвище,і ім’я, по батькові (підпис)

Консультант з графічної документації *доц., к.т.н., Ліщук К.І.* _____
посада,науковий ступінь,вчене звання,прізвище,і ім’я, по батькові (підпис)

Рецензент:

доц., к.т.н., доц. Ткач Михайло Мартинович _____
посада,науковий ступінь,вчене звання,прізвище,ім’я, по батькові (підпис)

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 року

UNICHECK

Власник документу:
Попенко Володимир Дмитрович

Дата перевірки:
11.06.2020 00:54:23 EEST

Дата звіту:
11.06.2020 22:24:18 EEST

ID перевірки:
1003947606

Тип перевірки:
Doc vs Internet + Library

ID користувача:
77149

Назва документу: Tsytsyliuk_ip61

ID **файлу:** 1003961331 **Кількість сторінок:** 53 **Кількість слів:** 8014 **Кількість символів:** 58165 **Розмір файлу:** 3.04 MB

9.93% Схожість

Найбільша схожість: 1.5% **з джерело бібліотеки.** ID **файлу:** 1000757531

3.58% Схожість з Інтернет джерелами	28	Page 55
8.29% Текстові збіги по Бібліотеці акаунту	51	Page 55

0% Цитат

Не знайдено жодних цитат

0% Вилучень

Вилучений текст відсутній

Підміна символів

Заміна символів	4

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет (інститут) *Інформатики та обчислювальної техніки*
(повна назва)

Кафедра *автоматизованих систем обробки інформації і управління*
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – *121 Інженерія програмного забезпечення*

Освітньо-професійна програма – *Програмне забезпечення інформаційних управляючих систем та технологій*

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ *Олександр ПАВЛОВ*
(підпис)
“___”_____2020 р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ СТУДЕНТУ

Цицилюк Анні Валеріївні
(прізвище, ім’я, по батькові)

1. Тема проєкту « Веб-застосування для виявлення аномалій в електрокардіограмах лінгвістичним методом »
керівник проєкту ст.викладач *Олійник Юрій Олександрович*

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “07” травня 2020 р. №1081-с

2. Термін подання студентом проєкту «08» *червня 2020 року*

3. Вихідні дані до проєкту

Технічне завдання

4. Зміст пояснювальної записки

1) Аналіз вимог до програмного забезпечення: основні визначення та терміни, опис предметного середовища, огляд існуючих технічних рішень та відомих програмних продуктів, розробка функціональних та нефункціональних вимог

2) Моделювання та конструювання програмного забезпечення: моделювання та аналіз програмного забезпечення, , архітектура програмного забезпечення, конст - руювання програмного забезпечення, математичне забезпечення

3)Аналіз якості та тестування програмного забезпечення

4)Впровадження та супровід програмного забезпечення

5. Перелік графічного матеріалу

1) Схема бази даних

2) Схема структурна класів програмного забезпечення

3) Креслення вигляду екранних форм

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання «10» *березня 2020 року*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	*Вивчення рекомендованої літератури*	*28.02.2020*	
2.	*Аналіз існуючих методів розв'язання задачі*	*07.03.2020*	
3.	*Постановка та формалізація задачі*	*12.03. 2020*	
4.	*Аналіз вимог до програмного забезпечення*	*17.03. 2020*	
5.	*Алгоритмізація задачі*	*25.03. 2020*	
6.	*Моделювання програмного забезпечення*	*29.03. 2020*	
7.	*Обґрунтування використовуваних технічних засобів*	*04.04. 2020*	
8.	*Розробка архітектури програмного забезпечення*	*19.04. 2020*	
9.	*Розробка програмного забезпечення*	*02.05. 2020*	
10.	*Налагодження програми*	*06.05. 2020*	
11.	*Виконання графічних документів*	*09.05.2020*	
12.	*Оформлення пояснювальної записки*	*22.05.2020*	
13.	*Подання ДП на попередній захист*	*29.05.2020*	
14.	*Подання ДП рецензенту*	*06.06.2020*	
15.	*Подання ДП на основний захист*	*08.06.2020*	

Студент _____ Анна ЦИЦИЛЮК
(підпис)

Керівник _____ Юрій ОЛІЙНИК
(підпис)

№з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
1	А4		Завдання на дипломний проєкт	2	
2	А4	КПІ.ІП-6125. 045420.01.81	Пояснювальна записка	72	
3	А4	КПІ.ІП-6125. 045420.02.91	Технічне завдання	10	
4	А4	КПІ.ІП-6125. 045420.03.34	Керівництво користувача	9	
5	А4	КПІ.ІП-6125. 045420.04.13	Опис програми	34	
6	А3	КПІ.ІП-6125. 045420.05.99 КЕ	Креслення вигляду екранних форм	1	
7	А3	КПІ.ІП-6125. 045420.05.99 СБД	Схема бази даних	1	
8	А3	КПІ.ІП-6125. 045420.05.99 СС	Схема структурна класів програмного забезпечення	3	

Зм.	Арк.	ПІБ	Підп.	Дата	КПІ.ІП-6125. 045420.00.90		
Розробн.		Цицилюк А.В			Відомість дипломного проєкту	Літ.	Лист 1 / Листів 1
Керівн.		Олійник Ю.О					
Консульт						КПІ ім.Ігоря Сікорського кафедра АСОІУ гр. ІП-61	
Н/контр.		Ліщук К.І.					
В.о.зав.каф.		Павлов О.А.					

АНОТАЦІЯ

Пояснювальна записка складається із 4 розділів та містить 11 рисунків, 26 таблиць, 6 додатків та 40 джерел – загалом 132 сторінки.

Метою проекту є розробка веб-застосування, основним призначенням якого є аналіз електрокардіограм та виявлення аномалій в них.

У розділі «Аналіз вимог до програмного забезпечення» проаналізована область дослідження, а саме описаний лінгвістичний метод , використані відстані , проведено аналіз успішних наукових робіт.

У розділі «Моделювання та конструювання програмного забезпечення» було розроблено архітектуру платформи, наведена діаграма класів та використані інструменти. В даному розділі описані основні використані технології та приведено ґрунтовний аналіз використаного архітектурного підходу.

У розділі «Аналіз якості та тестування програмного забезпечення» наведено приклади тестування програмного забезпечення та сам план тестування.

У розділі «Впровадження та супровід програмного забезпечення» описано основні кроки для успішного впровадження даного продукту та інструкція користувача для використання даного програмного забезпечення.

Ключові слова: ЛІНГВІСТИЧНИЙ АНАЛІЗ, ЕЛЕКТРОКАРДІОГРАМА, ЕКГ, СЕРЦЕВО-СУДИННІ ЗАХВОРЮВАННЯ,СИМВОЛЬНІ ЛАНЦЮГИ, АНОМАЛІЇ В ЕКГ.

ABSTRACT

The explanatory note consists of 4 sections and contains 11 figures, 26 tables, 6 appendices and 40 sources - a total of 132 pages.

The aim of the project is to develop a web application, the main purpose of which is the analysis of electrocardiograms and detection of anomalies in them.

In the section "Analysis of software requirements" the field of research is analyzed, namely the linguistic method is described, distances are used, the analysis of successful scientific works is carried out.

In the section "Software modeling and design" the platform architecture was developed, the class diagram is given and the tools are used. This section describes the main technologies used and provides a thorough analysis of the architectural approach used.

The section "Software Analysis and Software Testing" provides examples of software testing and the testing plan itself.

The section "Implementation and maintenance of software" describes the main steps for successful implementation of this product and user instructions for using this software.

Key words: LINGUISTIC ANALYSIS, ELECTROCARDIOGRAPHY, ECG, CARDIOVASCULAR DISEASES, SYMBOL CHAINS, ECG ANOMALIES.

Пояснювальна записка
до дипломного проєкту

на тему: Веб-додаток для виявлення аномалій в електрокардіограмах лінгвістичним методом _____

Київ – 2020 року

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

1) ЕКГ – електрокардіограма.
2) Фреймворк – програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту.
3) HTML – мова розмітки гіпертексту. Використовується для розробки веб-сайтів.
4) HTML-елемент – одиничний компонент HTML документу чи веб-сторінки.
5) JSON – мова розмітки, що визначає набір правил для представлення документів у форматі, що зрозумілий як комп’ютеру, так і людині.
6) API – це інтерфейс, що дозволяє двом незалежним компонентів програмного забезпечення обмінюватися інформацією. API грає роль посередника між внутрішніми і зовнішніми програмними функціями.
7) LDS – лінгвістичні динамічні системи.
8) CNN(Convolutional Neural Networks) – згорткова нейронна мережа
9) LSTM (long short-term memory) – архітектура нейронних мереж

ВСТУП

Серцево-судинні захворювання є широко розповсюдженими по всьому світу і являються однією з основних причин смерті. Кожен рік від серцево-судинних захворювань помирає 17,5 мільйонів людей по всьому світі. Аналіз Електрокардіограм дозволяє своєчасно визначити наявність різних відхилень в роботі серця таких як наявність блокад , пошкодження міокарду різної природи ішемічних (коли м'язі серця не вистачає кисню через порушення припливу крові по судинах) або дистрофічних (коли серце страждає вторинно через ендокринні порушення , зміні електролітного складу крові ,анемій та інших причин).Також аналіз Електрокардіограм дозволяє оцінити регулярність серцевого ритму(нормальний, занадто швидкий , повільний або нерегулярний). Саме завдяки правильній діагностиці, можливо вчасно виявити порушення серцевої діяльності та вжити всіх необхідних заходів для відновлення її нормального функціонування якнайшвидше.[1]

Тож наразі дуже актуально є розробка алгоритмів та програм для пришвидшення розшифровки та аналізу електрокардіограм.

Основною метою даної роботи є покращення виявлення аномалій на ЕКГ за рахунок використання лінгвістичного методу. Даний метод є швидким та не потребує великої кількості пам’яті.

Основним призначенням даного продукту є використання лінгвістичного методу для аналізу електрокардіограм і створення зручного інтерфейсу для завантаження даних електрокардіограм. Також цей проект має за мету вивчення можливості застосування лінгвістичних методів для знаходження аномалій і аналіз отриманих результатів.

Веб застосунок призначений для лікарів , які можуть завантажувати файли з даними ЕКГ та отримувати швидкий аналіз і звертати більше уваги на місця де була виявлена аномалія

Веб застосування було обрано через можливість мультиплатформеного застосунку та для швидкого та зручного використання іншими дослідниками зібраних даних (за рахунок API, яке буде описано далі).

1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Загальні положення

Для діагностики серцево-судинної системи людини найчастіше використовують саме аналіз ЕКГ тому що він є достатньо недорогим і показує значні результати з моменту його запровадження. Також важливою перевагою електрокардіографії є те що це абсолютно безпечний і безболісний метод діагностики. ЕКГ тобто його можна виконувати скільки завгодно раз через як завгодно короткі тимчасові проміжки.

Стандартна ЕКГ являє собою 12-векторне відображення електричної активності серця як відображення різниці електричних потенціалів між позитивними і негативними електродами, поміщеними на кінцівки і грудну клітку.[2]

В основному реєструють 6 грудних відведень : з V1 по V6. Відведення V7-V8-V9 незаслужено рідко використовуються в клінічній практиці, хоча вони дають більш повну інформацію про патологічних процесах в міокарді задньої (задньої-базальної) стінки лівого шлуночка. Основні грудні відведення ЕКГ описані в таблиці 1.1 та продемонстровані на рисунку 1.1.

Таблиця 1.1 – Опис розташування 6 грудних введень

Відведення	Розташування реєструючого електрода
V1	В 4-му міжребер'ї біля правого краю грудини
V2	В 4-му міжребер'ї біля лівого краю грудини
V3	На середині відстані між V2 і V4
V4	У 5-му міжребер'ї по середньо-ключичній лінії
V5	На перетині горизонтального рівня 4-го відведення і передній пахвовій лінії
V6	На перетині горизонтального рівня 4-го відведення і середній пахвовій лінії

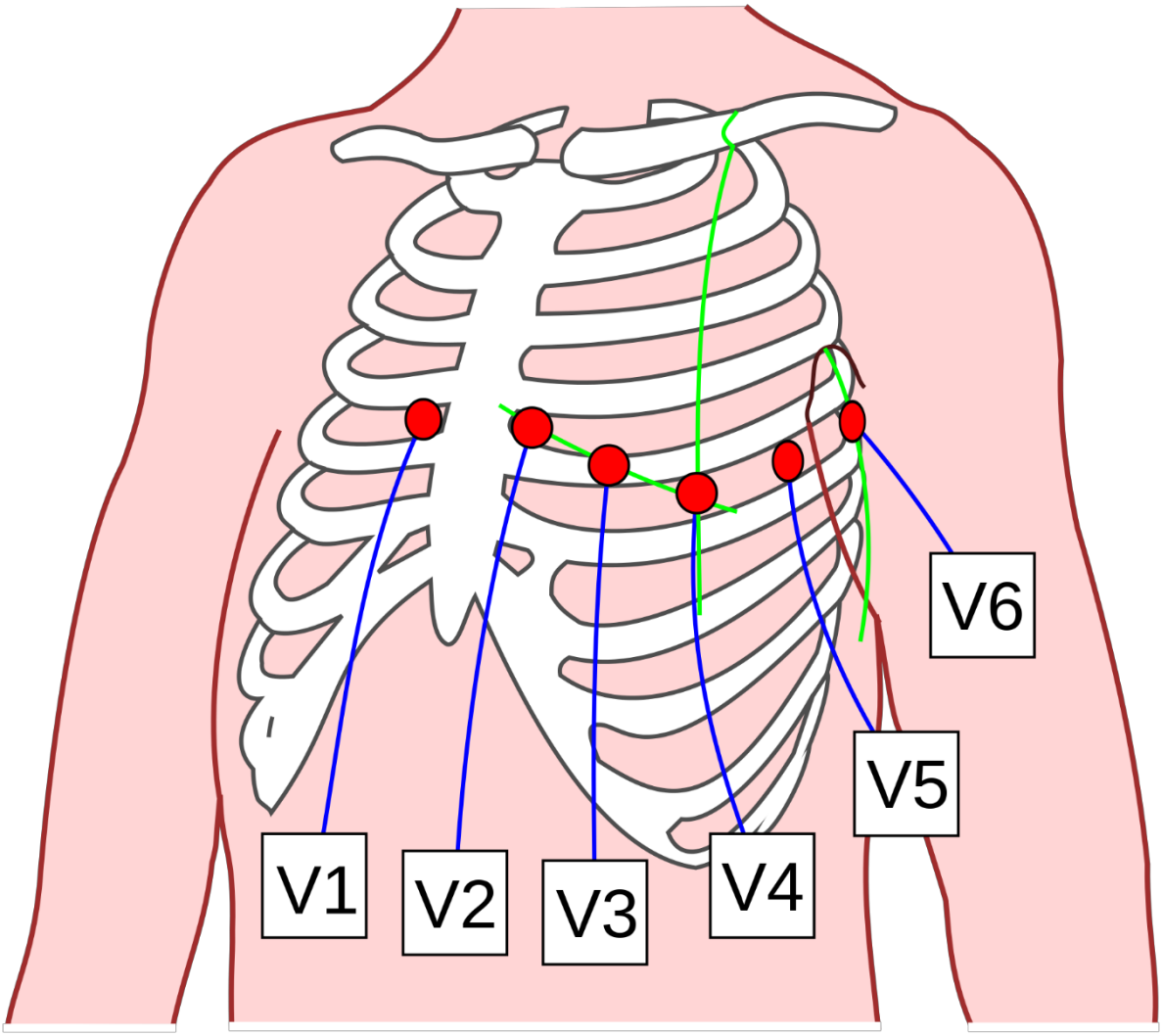


Рисунок 1.1 – Відображення 6-ти грудних введень

Загалом на виході отримуємо послідовність чисел на основі яких будується відповідне зображення ЕКГ , яке і буде проаналізоване спеціалістами. Загалом один цикл електрокардіограми здорової людини має відображається як показано на рисунку 1.2. Таким чином ми маємо шаблон який має постійно відтворюватись протягом всього циклу ЕКГ і тому відхилення від шаблонного рисунку на ЕКГ є помітними. Даний програмний продукт буде призначено для пошуку таких відхилень.

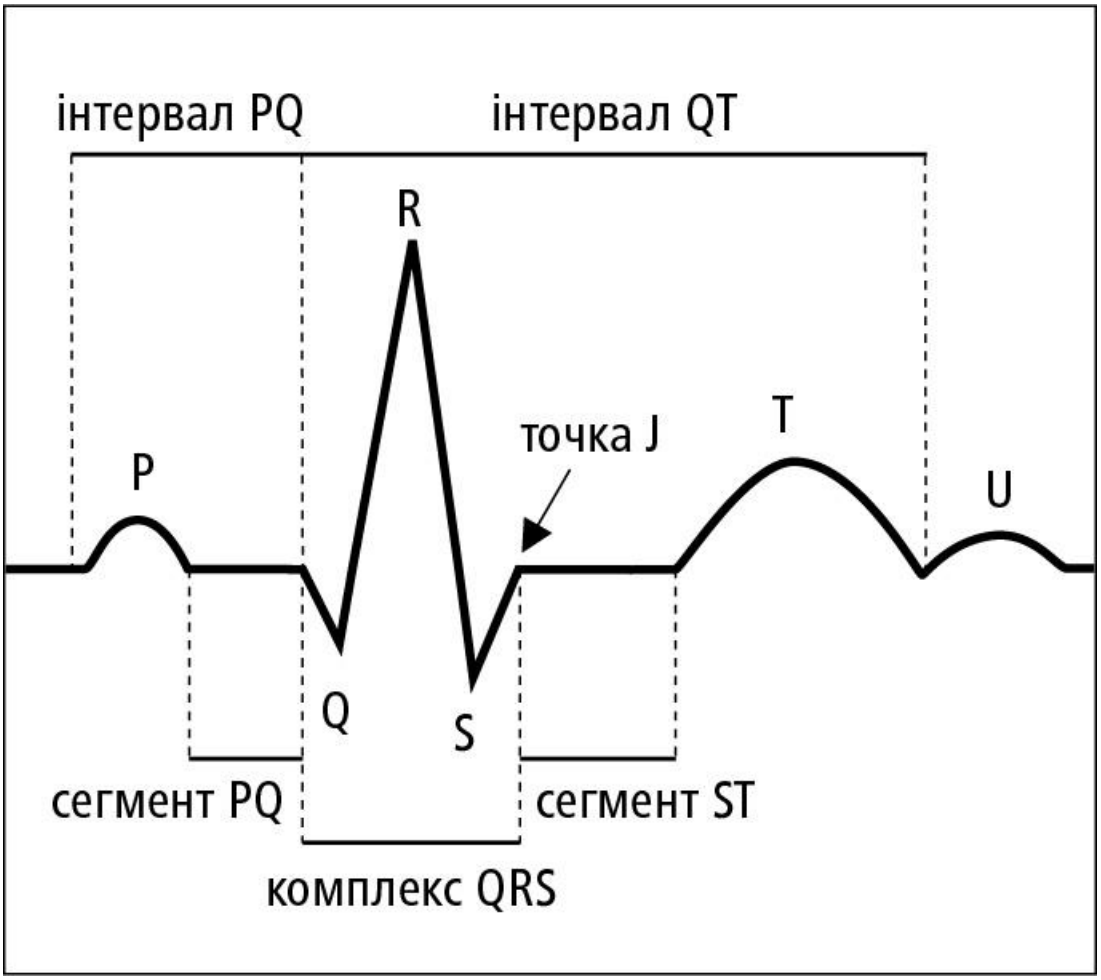


Рисунок 1.2 – Цикл серцевого ритму у здорової людини

1.2 Змістовний опис і аналіз предметної області

У середині 50-х років XX століття визначились основні принципи трактування лінгвістичних понять і математична лінгвістика почала бурхливо розвиватись. Прискоренню її розвитку істотно сприяла та обставина, що власне тоді, у зв’язку з появою обчислювальних машин і швидким зростанням потоку наукової інформації, була поставлена задача автоматизованого перекладу, інформаційного пошуку, побудова штучного здорового інтелекту та розпізнавання мови. Ці задачі привернули до лінгвістики увагу спеціалістів у галузі точних наук і сприяли розвитку цієї сфери.

Методології лінгвістичного моделювання відомі вже давно і можуть бути корисними в багатьох дослідженнях таких як біоінженерія(порівняння білків та інших сполук між собою), медицині (для знаходження хвороб опорно-рухового

апарату).Основним підходом при лінгвістичному методі є перетворення ряду дискретних даних (числового ряду) в символьний ланцюжок , де одному числу відповідає одна літера. Загалом лінгвістичне моделювання є спеціальним видом математичного тільки оперує літерами замість цифр.[3]

Опишемо вхідні дані для аналізу.

На вхід для аналітики дані можуть подаватись в багатьох форматах. Дана програма на вхід для бази даних з підтвердженими шаблонами аномалій отримує позначені дані з підтвердженням наявності аномалії та її типом. Приклад вхідних даних позначений на рисунках 1.3 та 1.4.

100

'sample #'	'MLII'	'V5'
0	995	1011
1	995	1011
2	995	1011
3	995	1011
4	995	1011
5	995	1011

Рисунок 1.3 – Формат потокових даних

Time	Sample #	Type
0:00.050	18	+
0:00.214	77	N
0:01.028	370	N
0:01.839	662	N
0:02.628	946	N
0:03.419	1231	N
0:04.208	1515	N
0:05.025	1809	N
0:05.678	2044	A
0:06.672	2402	N
0:07.517	2706	N
0:08.328	2998	N
0:09.117	3282	N
0:09.889	3560	N
0:10.728	3862	N

Рисунок 1.4 – Маркові дані для ЕКГ

На рисунку 1.4 зображений приклад даних, які ми отримуємо в результаті запису ЕКГ, час між такими записами проходить дуже малий приблизно 0,04 секунди – тому при записі 1 ЕКГ ми отримуємо близько 650000 таких екземплярів в одному файлів.

На рисунку 1.5 зображений приклад маркованих даних ЕКГ де , колонка “Time” відповідає за час з початку проведення замірів, “Sample #” – номер запису в 1 файлі (показаному на рисунку 1.4) , колонка “Type” відповідає за те є аномалія чи немає. Значення “N” в ній відображає відсутність аномалії , інші значення позначають певну серцеву аномалію.

Таким чином розглядаючи приклад показаний на рисунку 1.5 можемо зробити висновок про наявність аномалій на вимірі 18 на 2044 і на проміжку біля цих значень. При чому ці аномалії є різними.

У цій роботі дослідженні сигнали ЕКГ отримують із набору даних про аритмію MIT-BIH [4]. Набір даних MIT-BIH був основним набором традиційних експериментальних зразків для оцінки відхилень ЕКГ. Починаючи з 1980 р. Цей набір даних використовувався для початкового дослідження руху серця протягом понад 500 досліджень у всьому світі [5]. Цей набір даних містить 48

півгодинних витягів з подвійним контуром, цілодобові записи ЕКГ, отримані від 47 пацієнтів, які спостерігалися в лабораторії аритмії ВІН. Таким чином ми отримуємо достатню базу шаблонів для подальшого аналізу ЕКГ і виявлення аномалій.

Також ЕКГ в реальних даних може передаватись як файл з потоком значень, для подальшого аналізу в програмі.

Одним з можливих введень даних ЕКГ, яке і використовують лікарі є файли знімки ЕКГ такі які показані на рисунку 1.5 .

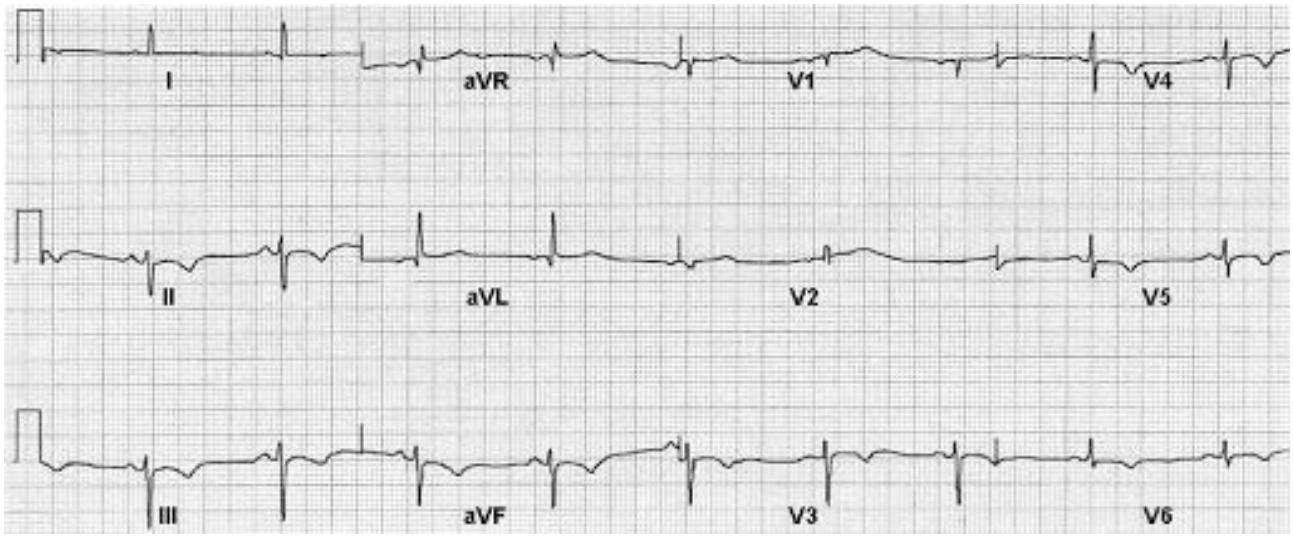


Рисунок 1.5 – Приклад фото ЕКГ

Опишемо загальний алгоритм обробки ЕКГ за допомогою лінгвістичного методу[6].

– Пошук точного шаблону відповідності з фрагментами в лінгвістичному ланцюжку.(Для чіткого знаходження аномалій).

– Пошук шаблону аномалій з певною похибкою відстані. Алгоритми використані для пошуку відстані були наведені раніше.

– Узагальнення сітки. Тобто зменшення або збільшення проміжків між вимірами. (Методами інтерполяції чи іншими).

– Повторення кроків 1-2.

– Корегування результатів по іншій осі(не часові а осі з результатами).

Зтиснення та розжимання результатів.

– Повторення кроків 1-2 .

На 3 кроці даного алгоритму проводиться розжимання або зжимання записів – тобто змінюється частота записів через те що різні апарати можуть вимірювати з різною частотою. Ми проводимо це за допомогою лінійної інтерполяції тому що це швидко(набагато швидше ніж більшість інших інтерполяцій) та частота точок є зазвичай достатньо великою – тому це дає змогу не втрачати данні і не отримати нові за рахунок заокруглень і тому подібного.

Корегування по самим значенням проводиться також з метою уніфікація всіх апаратів ЕКГ. Наведемо схему роботи даного алгоритму на рисунку 1.6.

Рисунок 1.6 – Структурна схема роботи аналізу ЕКГ

1.3 Аналіз успішних ІТ-проектів

1.3.1 Аналіз відомих технічних рішень

Оскільки дана робота має науковий напрям і одна з цілей вивчити ефективність лінгвістичного підходу при аналізі даних ЕКГ доцільно буде привезти аналіз наукових робіт і їх результатів.

Аналіз ЕКГ має велике значення в медицині і починає привертати більше уваги та брати участь у різних сферах. Наприклад, є дослідження [7] щодо ідентифікації людини в системі безпеки шляхом аналізу його ЕКГ. В цьому дослідженні був запропонований метод для обрізання інтервалів ЕКГ, в яких тренування нейронної мережі проводили, використовуючи кожен зразок даних зрізів як вхідний параметр.

Також була проведена робота [8] щодо створення веб-інструменту для поширених анотацій на ЕКГ. Ця кросплатформена система може використовуватися для спільного перегляду та анотації ЕКГ, що спростить спілкування між фахівцями. Також важливою складовою даного проекту є можливість маркування даних фахівцями для спрощення спілкування та для подальших досліджень отриманих даних за допомогою нейронних мереж та інших методів. За допомогою даного дослідження за 3 місяця було зібрано близько 15000 маркованих зразків.

Інший підхід до охорони здоров'я в реальному часі - це підхід у режимі реального часу, що базується на багатошаровому класифікаторі перцептрон [9]. На підставі цього відбувається розподіл вхідних серцевих скорочень ЕКГ на один з 23 класів, використовуючи рідко розподілені підписи ЕКГ. Проведені тести показали результат середню точність в 95,7% , крім того, розріджене представлення ЕКГ генерує лише 3,7% помилкових результатів.

Крім того, аналітики ЕКГ знайшли застосування у дослідженні сну та його стадій. Таким чином, використання ЕКГ та дихальних сигналів для визначення стадії сну за допомогою глибоких нейронних мереж було описано в [10]. Було

створено п'ять надточних нейронних мереж, кожна для різних типів дихання. Це дослідження може допомогти загалом при вивченні сну та процесів в організмі на різних його стадіях. Головною проблемою, з якою стикаються вчені в цьому дослідженні, було зниження точності внаслідок віку та / або більш важкої апное сну.

На сьогоднішній день проводиться чимало досліджень для вдосконалення аналізу та обробки даних ЕКГ. Взагалі нейронні мережі в основному використовуються для аналітики через її здатність обробляти складні і нечіткі дані.

При вивченні результатів ЕКГ існує кілька проблем - сторонній шум, обробка даних набором без втрати інформації для навчальних мереж та аналіз самої інформації.

По-перше, складні моделі глибокого навчання можуть бути корисними не тільки для аналізу даних, але і для усунення різних типів шуму (тобто блукання базового рівня, артефакт м’язів, артефакт руху електрода) [11]. На основі результатів тестів на зниження шуму між двома моделями DL (CNN, LSTM). CNN мав найкращий показник як в синтетичних, так і в реальних даних. Тому дослідження найкращої та найпродуктивнішої моделі CNN в тому ж дослідженні стає актуальним.

По-друге, для аналізу даних у більшості випадків записи однакової довжини є неодмінними - відповідно існує кілька методів приведення записів до однакового розміру.

- Заповнення всіх записів до найбільшої довжини.
- Скорочення всіх записів до найкоротшого.
- Групування записів за довжиною.
- Обрізання даних на певну довжину [12].
- Обрізання даних на основі евристики [13].

Для самого аналізу описано велику кількість нових методів. Так, у [14] описаний алгоритм кластеризації часових рядів, який підходить для

багатовимірних входів і виходів змінної довжини та зміщення часу. Ця кластеризація заснована на відстані між прихованими лінійними динамічними системами (LDS). Приклад використання методу критичного коливання для аналізу ЕКГ був проведений у роботі[15], де вибір сегментів ЕКГ проводився за критерієм стаціонарності. Даний метод здатний виявити критичні характеристики, з точки зору фізичної поведінки, в експериментально записаних сигналах. Основним недоліком даного підходу що ця робота тлумачить роботу серця з точки зору фізичних критичних явищ і не вивчає медичну фізіологію

У [16] було винайдено новий інструмент для отримання нової інформації, який базується на новому методі аналізу даних - Різноманітне навчання. Ця методика витягує структуру з даних і належить до неконтрольованих методів машинного навчання. Тож із необроблених даних можна окреслити динаміку процесу всередині і з'являється можливість 3D-візуалізації DM. (Де кожен імпульс - точка)

В іншому дослідженні [17] було доведено, використовуючи мережі рецидивів, що RN з ЕКГ f є бімодальними і, таким чином, утворює окремий клас. Крім того, це дослідження показує, що РНК ЕКГ мають значно вищу величину для коефіцієнта кластеризації та нижчу величину для середньої довжини шляху.

Зрештою, можна сказати, що для ЕКГ-аналітики важливо зберегти дані при зміні часу запису, усунути зайвий шум для корекційного аналізу та оптимізувати алгоритми тренувань та покращити сам аналіз. результати досліджень показали, що сучасними підходами до аналізу ЕКГ є конволюційні нейронні мережі, мережі рецидивів, часові ряди, метод критичних коливань та нові методи навчання нейронних мереж (наприклад, різноманітні тренування). Найбільшою перевагою нейронних мереж є можливість аналізу нових даних на основі узагальнення попередніх випадків, але в той же час це і недолік, оскільки для навчання потрібно багато даних та часу.

1.3.2 Аналіз відомих програмних продуктів

Є багато платних сервісів , які пропонують послуги лікарів – тобто вони просто пересилають файл ЕКГ від пацієнта до лікаря і лікар вже аналізує отримане ЕКГ.

Більшість програм для аналізу ЕКГ становляться на самі апарати та зазвичай коштують достатньо дорого.

До того ж важливо зазначити , що вони не відкривають свого програмного коду , а зберігають його як підприємницьку таємницю .Також точно не можна сказати їх продуктивність та методи які вони використовують.

1.4 Аналіз вимог до програмного забезпечення

Програмне забезпечення надає користувачу можливість завантажити дані ЕКГ та отримати результат її аналізу лінгвістичним методом. Результат буде описаний текстово та продемонстровано в вигляді графіку.

Також важливою складовою даного продукту є авторизація, аутентифікація та збереження приватних даних користувача(оскільки всі медичні дані є приватними).

Оскільки пошта є зручним форматом для отримання результатів, важливою складовою функціоналу є отримання поштового сповіщення про результати та сама реєстрація поштової скриньки з підтвердженням.

1.4.1 Розроблення функціональних вимог

Для всіх можливих функцій програми продемонструємо таблицю з ідентифікатором, описом, актором, передумовою та вихідними умовами.

Таблиця 1.2 – Варіант вимог «Перегляд основної сторінки сайту»

Ідентифікатор	UC01
Назва	Перегляд основної сторінки сайту
Актор	Гість/Користувач

Продовження таблиці 1.2

Опис	-
Попередні умови	-
Вихідні умови	-

Таблиця 1.3 – Варіант вимог «Реєстрація»

Ідентифікатор	UC02
Назва	Реєстрація
Актор	Гість
Опис	Гість може створити свою сторінку
Попередні умови	-
Вихідні умови	Збереження даних про користувача та створення його сторінки

Таблиця 1.4 – Варіант вимог «Вхід»

Ідентифікатор	UC03
Назва	Вхід
Актор	Гість
Опис	Гість може зайти на свою сторінку
Попередні умови	Гість попередньо виконував операцію реєстрації
Вихідні умови	Гість переходить в стан користувача

Таблиця 1.5 – Варіант вимог «Аналіз ЕКГ»

Ідентифікатор	UC04
Назва	Аналіз ЕКГ
Актор	Гість/Користувач/Сторонній сервіс через API

Продовження таблиці 1.5

Опис	Актор може запросити аналіз даних ЕКГ і отримати їх в графічному та текстовому вигляді. Також Користувач може додатково їх зберегти на свою сторінку та відправити на пошту
Попередні умови	-
Вихідні умови	Аналіз переданої ЕКГ

Таблиця 1.6 – Варіант вимог «Додання пошти»

Ідентифікатор	UC05
Назва	Додання пошти
Актор	Користувач
Опис	Актор може додати пошту до свого облікового запису.
Попередні умови	Гість попередньо виконував операцію реєстрації
Вихідні умови	Пошта прив’язана до облікового запису користувача і результати обробки ЕКГ можуть бути швидко надіслані на неї

Таблиця 1.7 – Варіант вимок «Відновлення паролю»

Ідентифікатор	UC06
Назва	Відновлення паролю
Актор	Гість
Опис	Актор може відновити пароль за допомогою пошти , при вказанні пошти або логіну користувача.
Попередні умови	Пошта прив’язана до облікового запису гостя
Вихідні умови	На пошту буде Надісланий новий пароль користувача.

1.4.2 Розроблення нефункціональних вимог

Важливою складовою є отримання API для інших програм для подальших досліджень цієї галузі.

Також потрібним функціоналом є перетворення вхідних значень та запис отриманих даних в окремий файл для можливості провадження додаткової аналітики сторонніми програмами.

Найбільш затратною частиною даної програми є пам’ять тому що один запис ЕКГ – 650000 рядків з мінімальними даними. При зростанні кількості користувачів кількість інформації яку потрібно зберігати постійно зростатиме. Тож для нормального функціонування додатки потрібно щонайменше 1 ТБ пам’яті на сервері.

1.4.3 Постановка комплексу завдань

Мета – розробка застосунку для аналізу ЕКГ за допомогою лінгвістичного методу.

Призначення – аналіз ЕКГ для пошуку аномалій в них для попередження про можливі серцеві хвороби.

Задачі – збір тестових даних ЕКГ , вибір метрик відстані для лінгвістичного аналізу ЕКГ , створення клієнтської частини для взаємодії з алгоритмом, реалізація алгоритму для лінгвістичного аналізу ЕКГ.

Опишемо детальніше кожну з задач.

Збір тестових даних – реалізація парсинга (аналізу вхідних даних на наявність певних елементів) вхідних даних для зібрання інформації про типи аномалій та їх шаблони для подальшого аналізу.

Вибір метрики відстані для лінгвістичного аналізу – проведення досліджень та тестів для вибору найбільш корисних метрик для виміру відстаней між поточними даними та шаблонами аномалій

Створення клієнтської частини для взаємодії з алгоритмом – створення веб-застосунку для отримання даних для подальшого аналізу за допомогою лінгвістичного підходу.

Реалізація алгоритму для лінгвістичного аналізу ЕКГ – розробка та тестування алгоритму з лінгвістичним методом.

1.5 Висновки по розділу

Була проаналізована специфіка предметної області аналізу ЕКГ і приведені основні методи її аналізу. Найбільш розповсюдженим є метод визначення аномалій за допомогою різних типів нейронних мереж.

Основною новизною в більшості досліджень є модифікація методів навчання мережі та різні підходи до модифікації вхідних значень ЕКГ (шаблонізація їх в розмірі).

Також було проаналізоване лінгвістичне моделювання, його актуальність і галузі в яких воно застосовується(біоінформатика , аналіз тексту , зчитування образів і їх розпізнання, аналіз різних числових рядів , штучний переклад тексту).

Можемо зробити висновок на основі даних про відомі рішення для аналізу ЕКГ та даних про застосування лінгвістичного аналізу, що дана робота є унікальним рішенням для аналітики ЕКГ, важливим параметром якого є сама перевірка можливості його використання.

2 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Моделювання та аналіз програмного забезпечення

За допомогою технології BPMN було продемонстровано основний процес використання даного застосунку, а саме аналіз ЕКГ.

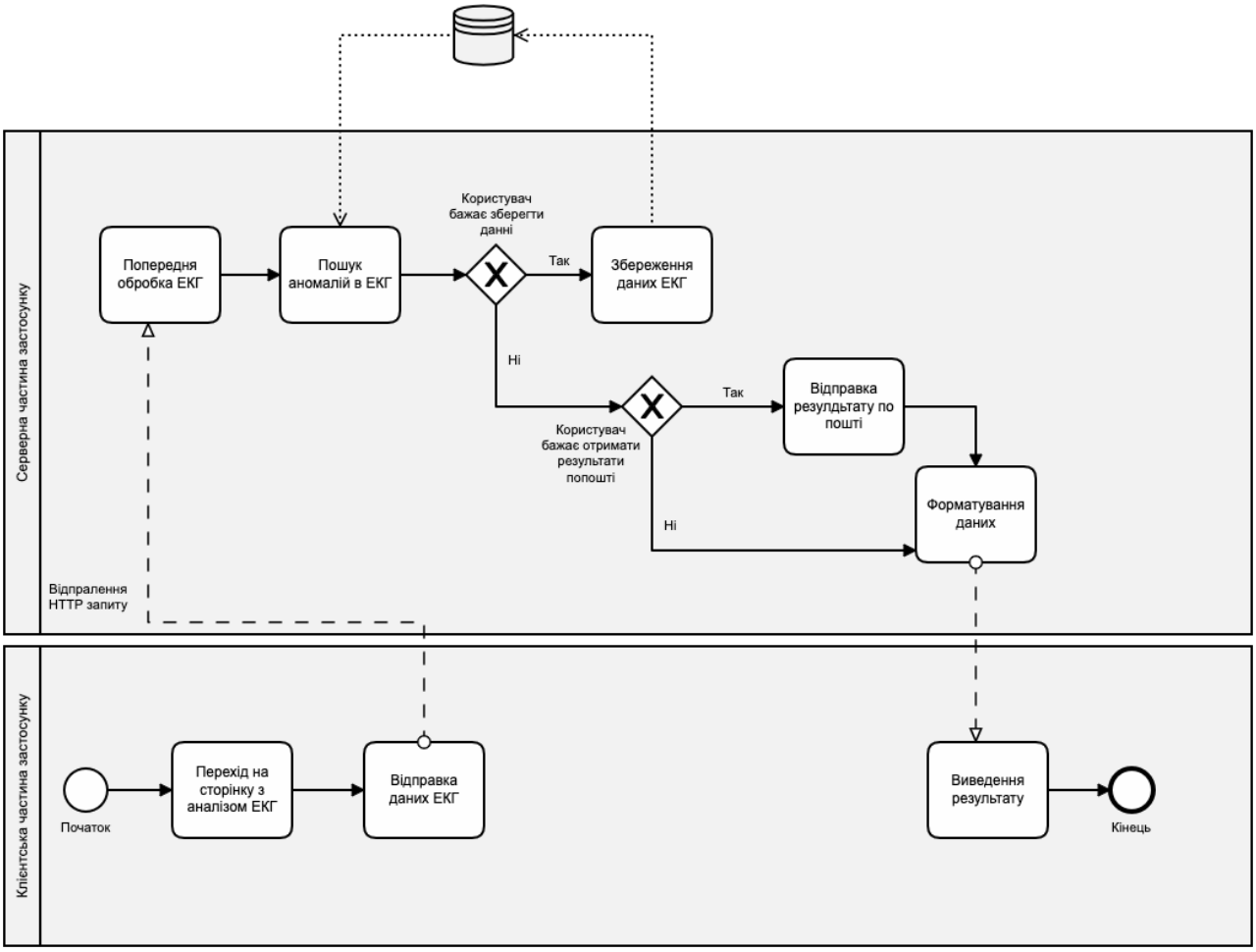


Рисунок 2.1 – Схема бізнес-процесу

Послідовність кроків для основного бізнес процесу в застосунку :

– перехід на сторінку з аналізом ЕКГ;
– відправка даних ЕКГ;
– попередня обробка ЕКГ;
– отримання шаблонів аномалій з бази даних ;
– аналіз ЕКГ;
– збереження результатів;
– відправлення результатів на пошту;
– показ даних користувачеві.

2.2 Архітектура програмного забезпечення

Основним підходом до структуризації коду був вибраний патерн MVC. Основні складові наведені на рисунку 2.1. [18]

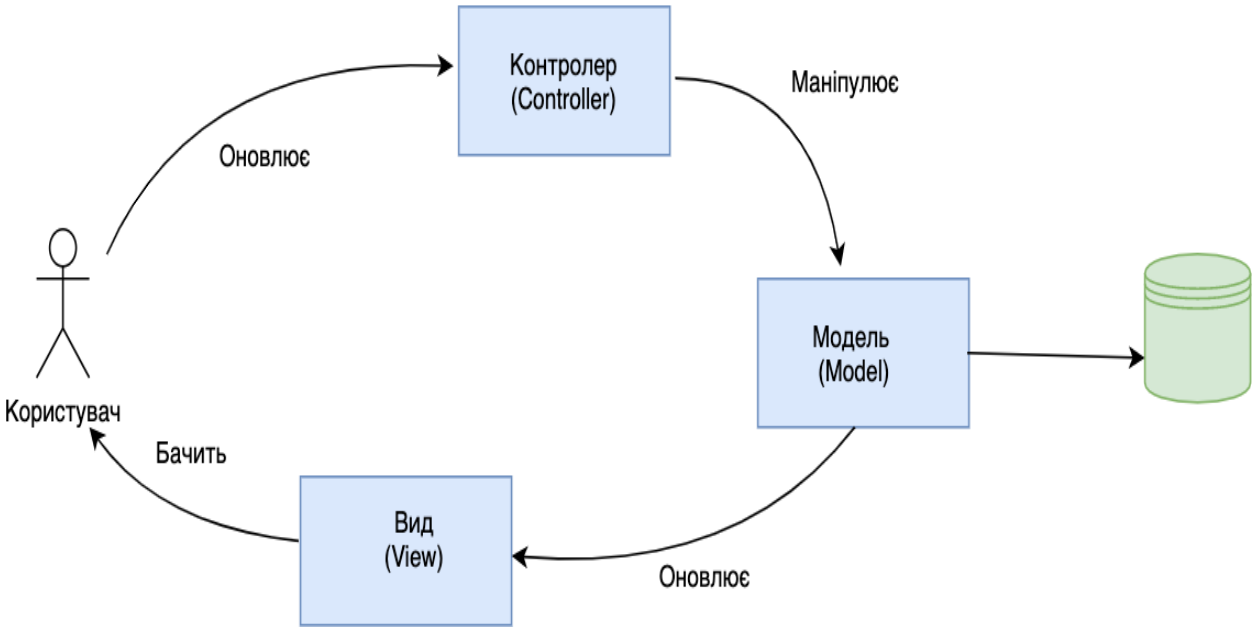


Рисунок 2.2 – Архітектура MVC

Три важливі MVC компоненти:

– модель: Вона включає всі дані та пов'язану з цим логіку;
– вид: представлення даних користувачеві або обробка взаємодії з користувачем;

– контролер: інтерфейс між компонентами Модель і Вид.

Опишемо кожну з них детальніше

Вид – це частина програми, яка представляє представлення даних.

Види створюються за даними, зібраними з даних моделі. Вид вимагає, щоб модель надавала інформацію так, щоб вона нагадувала вихідну презентацію користувачеві.

Вид також представляє дані з чатів, діаграм та таблиці. Наприклад, будь-який перегляд клієнта буде включати всі компоненти інтерфейсу користувача, наприклад текстові поля, вкладені поля тощо.

Контролер – це та частина програми, яка обробляє взаємодію з користувачем. Контролер інтерпретує введення миші та клавіатури від користувача, інформуючи модель та вигляд, щоб змінити, у відповідних випадках.

Контролер надсилає команди моделі для оновлення свого стану (наприклад, збереження конкретного документа). Контролер також надсилає команди до пов'язаного з ним перегляду для зміни подання перегляду (наприклад, прокрутка певного документа).

Компонент моделі зберігає дані та пов'язану з цим логіку. Він представляє дані, що передаються між компонентами контролера або будь-якою іншою пов'язаною бізнес-логікою. Наприклад, об'єкт Controller отримає інформацію про клієнта з бази даних. Він маніпулює даними та надсилає назад до бази даних або використовує їх для надання тих же даних.

Він відповідає на запит від видів, а також відповідає на вказівки контролера щодо оновлення. Це також найнижчий рівень структури, який відповідає за збереження даних.

Основні переваги використання моделі MVC:[19]

– легке обслуговування коду легко розширювати та нарощувати;

– компонент моделі MVC можна перевірити окремо від користувача;

– простіша підтримка нового типу клієнтів(під клієнтом розуміємо певний вид програмного забезпечення);

– розробка різних компонентів може виконуватися паралельно;

– це допомагає уникнути складності, розділивши додаток на три одиниці;

– він використовує лише шаблон переднього контролера, який обробляє запити веб-додатків через один контролер;

– забезпечує чітке розділення проблем ;

– усі класифіковані об'єкти не залежать один від одного, так що можливо перевірити їх окремо;

– MVC дозволяє логічно групувати пов'язані дії на контролері разом.

Опишемо використаний фреймворк для роботи, а саме Spring та Spring Boot. Spring, мабуть, найкраща з компонентних каркасів, що з’явилися на зламі 21 століття. Це значно покращує спосіб розробників записувати та робити інфраструктурний код у Java-додатках. З моменту свого створення, Spring визнана провідною основою для розвитку Java. Як основна програма застосувань "Spring" відображає деякі можливості Java EE [20], але він пропонує поєднання функцій та умов програмування, яких ви не знайдете в іншому місці.

Основна ідея Spring полягає в тому, що замість того, щоб самостійно керувати об’єктними відносинами, ви завантажуєте їх у фреймворк. Інверсія управління - це методологія, що використовується для управління відносинами об'єктів. Ін'єкція залежностей – це механізм реалізації інверсії залежностей. Оскільки ці два поняття пов'язані, але різні, давайте розглянемо їх детальніше.[21]

Інверсія управління робить саме те, що говорить назва: вона перевертає традиційну ієрархію управління для виконання об'єктних відносин. Замість того, щоб покладатися на код програми, щоб визначити, як об'єкти співвідносяться один з одним, відносини визначаються рамкою. Як методологія, вводить послідовність і передбачуваність об'єктних відносин, але це вимагає від вас, як розробника, відмовитися від тонкого контролю.

Введення залежності (DI) – це механізм, коли фреймворки "вводять" залежності у вашу програму. Це практична реалізація МОК. Ін'єкційна залежність залежить від поліморфізму, в тому сенсі, що вона дозволяє виконувати еталонний тип для зміни на основі конфігурацій в фреймворку. Фреймворку вводить змінні посилання, а не встановлювати їх вручну в коді програми.

Переваги Spring:[22]

– Spring дає можливість розробникам розробляти додатки корпоративного класу за допомогою POJO. Перевага використання лише POJO полягає в тому, що вам не потрібен контейнерний продукт EJB, такий як сервер додатків, але у вас є можливість використовувати лише надійний контейнер сервлетів, такий як Tomcat або якийсь комерційний продукт;

– Spring організований модульно. Незважаючи на те, що кількість пакетів та класів значна, вам потрібно турбуватися лише про потрібні, а решта ігнорувати;

– тестування програми, написаної за допомогою Spring, є простим. Крім того, за допомогою POJOs JavaBeanstyle стає легше використовувати ін'єкцію залежності для введення даних тесту;

– веб-рамка Spring - це добре розроблена веб-структура MVC;

– Spring пропонує зручний API для перекладу винятків, що стосуються технологій (наприклад, JDBC, Hibernate або JDO) у послідовні неперевірені винятки;

– Spring пропонує послідовний інтерфейс управління транзакціями, який може масштабувати до локальної транзакції (наприклад, за допомогою єдиної бази даних) і масштабувати до глобальних транзакцій (наприклад, використовуючи JTA).

Опишемо використаний фреймворк для роботи з даними а саме Hibernate.[23]

Hibernate – це ORM фреймворк для Java з відкритим кодом. Ця технологія є крайньою потужністю і має високі показники продуктивності.

Hibernate створює зв'язок між таблицями у базі даних та Java-класами та наборотом. Це позбавляє розробників від величезної кількості зайвої , рутинної роботи, в якій доволі легко зробити помилку і дуже важко потом знайти її.

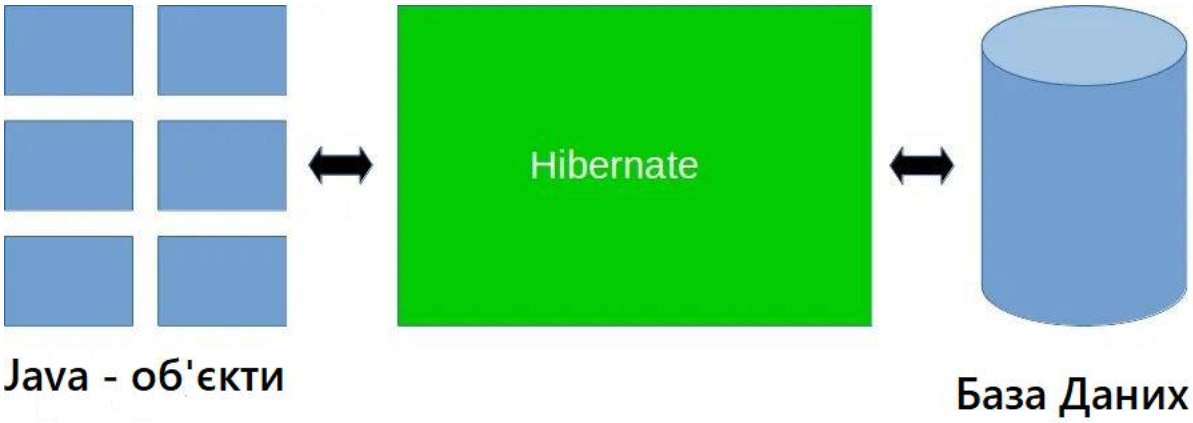


Рисунок 2.3 – Архітектура Hibernate

Переваги Hibernate[24]:

– забезпечує простий API для запису та отримання Java-об'єктів в / з БД;

– мінімізує доступ до БД, використовуючи стратегії fetching;

– не вимагає сервера додатка;

– дозволяє не працювати з типами даних мови SQL, а мати справу з звичний нам типами даних Java;

– піклується про створення зв'язків між Java-класами і таблицями БД за допомогою XML-файлів не вносячи зміни в програмний код;

– якщо нам неоходімо змінити БД, то достатньо лише внести зміни в XML-файли.

Розглянемо архітектуру Hibernate.

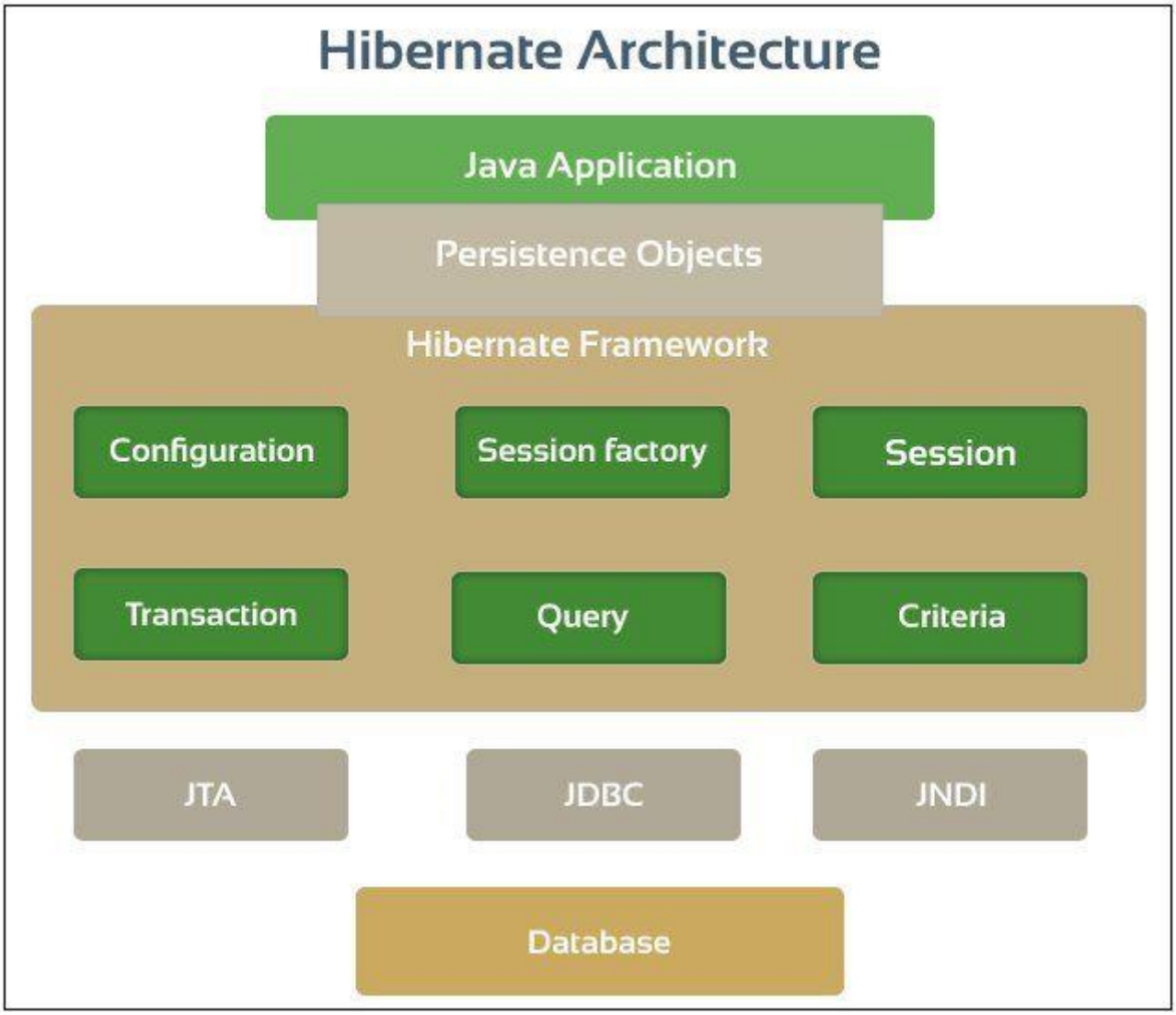


Рисунок 2.4 – Архітектура Hibernate

Transaction - цей об'єкт представляє собою робочу частину роботи з БД. У сплячому режимі транзакції обробляються менеджером транзакцій.[25]

SessionFactory – Самий важливий і найбільший об’єкт (зазвичай створюється в єдиному екземплярі, при запуску). Нам потрібно як мінімум одна SessionFactory для кожної БД, кожна з яких налаштовується окремим конфігураційним файлом.

Session – використовується для отримання фізичних з'єднань з БД. Зазвичай, сесія створюється при необхідності, а після цього закривається. Це пов’язано з тем, що є об'єкт сесії є дуже легким. Для того, щоб зрозуміти, що це

таке, можна сказати, що створення, читання, зміна і видалення об'єкта відбувається через об'єкт Сесія.

Query – Цей об'єкт використовує HQL або SQL для читання / запису даних з / в БД. Екземпляр запита використовується для зв'язки параметрів запиту, обмежуючої кількість результатів, які будуть повернуті при виконанні запиту.[26]

Configuration – Цей об'єкт використовується для створення об'єкта SessionFactory і налаштовує сам сплячку за допомогою коніфігураційного XML-файлу, який містить об'єкт, як створює об'єкт Session.[27]

Criteria – Використовується для створення і виконання об'єктів-орієнтованих запиту для отримання об'єктів.

Опишемо фреймворк Angular для роботи з клієнтом – тобто для відображення потрібної інформації на сторінках браузера.[28]

– З допомогою даного фреймворку спрощується написання зовнішнього вигляду для веб сторінок.

– Збільшується модульність застосунку.

– В фреймворку є багато вбудованих сервісів.

– Універсальний фреймворк для будь-якої серверної частини частини.

2.3 Конструювання програмного забезпечення.

Для бази даних була обрана ряляційна база даних з відкритим кодом PostgreSQL[29], через те що вона є безкоштовною , гарно оптимізованою та постійно розвивається. Опишемо створену базу даних для отриманого продукту показану в відповідному додатку.

Проаналізуємо основні таблиці.

Таблицею, яка в майбутньому займе багато місця якщо вона буде постійно заповнюватись і ці данні будуть використовуватись – Example. Загалом заповнення цієї таблиці не є обов’язковим і при заповнені таблиці з результатами не несе великого інформаційного навантаження.

Таблиця 2.1 – Опис таблиці Example

Поле	Тип	Первинний ключ	Обов'язковість	Призначення
Id	integer	Так	Так	Унікальний ідентифікатор запису
Mlii	integer	Ні	Так	Перше значення отриманої ЕКГ(по якому і проводиться наша аналітика)
V5	integer	Ні	Так	Друге значення отриманої ЕКГ
Example_time	integer	Ні	Ні	Час отриманого результату
Letter	integer	Ні	Так	Символ, який відповідає «першому» числовому значенню з ЕКГ
Result_id	integer	Ні	Ні	Унікальний ідентифікатор результуючого запису про дане ЕКГ.

Наступною таблицею, яку потрібно описати є таблиця-словник з типами аномалій. Опишемо спочатку можливі значення в даній таблиці.

Таблиця 2.2 – Опис символів типів аномалій

Символ в вхідних даних	Ідентифікатор
""	0
"+"	1
"N"	2
"A"	3
"~"	4

Продовження таблиці 2.2

"|"	5
"V"	6

Таблиця 2.3 – Опис таблиці Type of anomaly

Поле	Тип	Первинний ключ	Обов'язковість	Призначення
Id	integer	Так	Так	Унікальний ідентифікатор запису
Type_of_anomaly	text	Ні	Так	Назва аномалії в ЕКГ

Наступною сутністю для опису буду Такт – це один крок циклу ЕКГ (тобто з чого складається сама електрокардіограма). Він введений в поточну база даних для подальшого розширення але наразі не є обов’язковим до заповнення.

Таблиця 2.4 – Опис таблиці Tact

Поле	Тип	Первинний ключ	Обов'язковість	Призначення
Id	integer	Так	Так	Унікальний ідентифікатор запису
Start_time	integer	Ні	Ні	Початок запису результатів певного такту
End_time	integer	Ні	Ні	Кінець запису результатів певного такту
Tact_string	text	Ні	Так	Результуючий рядок з даними такту переведеними в певні літери
Result_id	integer	Ні	Ні	

Далі доцільно описати саму таблицю з аномалією в електрокардіограмі. Варто зауважити про наявність 2 таблиць в базі даних для різних аномалій. 1 таблиця призначена для запису позначених аномалій (як шаблонних). 2 таблиця призначена для запам’ятовування знайдених аномалій і позначення впевненості в них.

Таблиця 2.5 – Опис таблиці Anomaly

Поле	Тип	Первинний ключ	Обов'язковість	Призначення
Id	integer	Так	Так	Унікальний ідентифікатор запису
Anomaly_string	text	Ні	Так	Рядок з аномалією , який містить
Anomaly_type_id	integer	Ні	Ні	Рядок з ідентифікатором типу аномалії

Таблиця 2.6 – Опис таблиці PredictedAnomaly

Поле	Тип	Первинний ключ	Обов'язковість	Призначення
Id	integer	Так	Так	Унікальний ідентифікатор запису
Anomaly_string	text	Ні	Так	Рядок з аномалією , який містить
Anomaly_type_id	integer	Ні	Ні	Рядок з ідентифікатором типу аномалії
Probability	double	Ні	Так	Показує вірогідність того , що даний запис є аномалією.

Оскільки попередньо були описані всі складові результату аналізу ЕКГ, можемо перейти до його опису.

Таблиця 2.7 – Опис таблиці Result

Поле	Тип	Первинний ключ	Обов'язковість	Призначення
Id	integer	Так	Так	Унікальний ідентифікатор запису
Result_string	text	Ні	Так	
Anomaly	integer	Ні	Так	Кількість виявлених аномалій
Person_id	integer	Ні	Ні	Унікальний ідентифікатор людини , яка внесла наступні данні ЕКГ

Для повноцінної роботи веб застосунку і роботи з користувачами а також для зібрання медичних даних для подальшого аналізу необхідні таблиці для заповнення його даних. Дані користувача було розподіллено на 2 таблиці – дані з інформацією для роботи з веб застосунком і дані зібрані в медичних цілях.

Таблиця 2.8 – Опис таблиці Person

Поле	Тип	Первинний ключ	Обов'язковість	Призначення
Id	integer	Так	Так	Унікальний ідентифікатор запису
Age	integer	Ні	Ні	Вік користувача
Weight	integer	Ні	Ні	Вага користувача
Height	integer	Ні	Ні	Зріст користувача
Sex	integer	Ні	Ні	Стать користувача

Таблиця 2.9 – Опис таблиці User

Поле	Тип	Первинний ключ	Обов'язковість	Призначення
Id	integer	Так	Так	Унікальний ідентифікатор запису
Login	text	Ні	Так	Логін користувача(унікальний ідентифікатор в системі, який видний користувачеві)
Password	text	Ні	Так	Пароль користувача(в шифрованому вигляді)
Email	text	Ні	Ні	Поштова скринька
Is_email_ verified	bool	Ні	Ні	

Опишемо основні класи серверної частини програми.

Таблиця 2.10 – Опис основних об’єктів в програмі.

Назва	Тип	Пакет	Призначення
DamerauLevenshtein	Клас	Algorithm	Клас для реалізації алгоритму виміру відстані Дамерау-Левінштейна
XCompression	Клас	Algorithm	Клас для зжимання та розжимання даних ЕКГ по часовому ряду
YCompression	Клас	Algorithm	Клас для зжимання та розжимання даних ЕКГ по осі значень.

Продовження таблиці 2.10

Compression	Інтерфейс	Algorithm	Інтерфейс для зжимання та розжимання даних
AnomalyDAO	Інтерфейс	DAO	Інтерфейс для роботи з моделлю аномалії ЕКГ.
ExampleDAO	Інтерфейс	DAO	Інтерфейс для роботи з моделлю запису ЕКГ.
ResultDAO	Інтерфейс	DAO	Інтерфейс для роботи з моделлю запису ЕКГ.
TactDAO	Інтерфейс	DAO	Інтерфейс для роботи з моделлю такту ЕКГ.
PersonDAO	Інтерфейс	DAO	Інтерфейс для роботи з моделлю користувача .
AnomalyDAOImpl	Клас	DAO.impl	Базова реалізація інтерфейсу AnomalyDAO для роботи з моделлю аномалії ЕКГ.
ExampleDAOImpl	Клас	DAO.impl	Базова реалізація інтерфейсу ExampleDAO для роботи з моделлю запису ЕКГ.

Продовження таблиці 2.10

ResultDAOImpl	Клас	DAO.impl	Базова реалізація інтерфейсу ResultDAO для роботи з моделлю результату ЕКГ.
TactDAOImpl	Клас	DAO.impl	Базова реалізація інтерфейсу TactDAO для роботи з моделлю такту ЕКГ.
HibernateSessionFactoryUtil	Клас	DAO	Клас для налаштування конфігурації Hibernate сесії
FileType	Перелік	FileSystem	Перелік для можливих типів(розширень) вхідного файлу. Можливі значення: Txt, Csv
InputFile	Клас	FileSystem	Клас для опису вхідного типу файлу , який містить назву та розширення файлу
SimpleFileReader	Клас	FileSystem	Клас який реалізує порядкову обробку вхідних файлів.

Продовження таблиці 2.10

Anomaly	Клас	Model	Клас який служить відображенням моделі даних аномалії і проводить співвідношення з таблицями в базі даних за допомогою анотацій
AnomalyType	Перелік	Model	Перелік ,який відповідає таблиці-словнику з типами аномалій ЕКГ , для подальшого використання в програмі.
Example	Клас	Model	Клас який служить відображенням моделі даних запису і проводить співвідношення з таблицями в базі даних за допомогою анотацій

Продовження таблиці 2.10

Person	Клас	Model	Клас який служить відображенням моделі даних користувача і проводить співвідношення з таблицями в базі даних за допомогою анотацій
Result	Клас	Model	Клас який служить відображенням моделі даних результату і проводить співвідношення з таблицями в базі даних за допомогою анотацій
Tact	Клас	Model	Клас який служить відображенням моделі даних такту і проводить співвідношення з таблицями в базі даних за допомогою анотацій
AnomalySearcher	Інтерфейс	Services	Інтерфейс для пошуку аномалій ЕКГ в рядку значень.

Продовження таблиці 2.10

AnomalyService	Інтерфейс	Services	Інтерфейс для роботи з аномаліями(пошук, збереження , видалення)
ExampleService	Інтерфейс	Services	Інтерфейс для роботи з записами(пошук, збереження , видалення)
LinguisticChainService	Інтерфейс	Services	Інтерфейс для узагальнення роботи з файлами і отримання аномалій
ResultService	Інтерфейс	Services	Інтерфейс для роботи з результатами(пошук, збереження , видалення)
TactService	Інтерфейс	Services	Інтерфейс для роботи з тактами(пошук, збереження , видалення)
AnomalySearcherImpl	Клас	Services.impl	Базова реалізація інтерфейсу AnomalySearcher для пошуку аномалій в рядку

Продовження таблиці 2.10

AnomalyServiceImpl	Клас	Services.impl	Базова реалізація інтерфейсу AnomalyService для реалізації роботи з аномаліями(пошук, збереження , видалення)
ExampleServiceImpl	Клас	Services.impl	Базова реалізація інтерфейсу ExampleService для реалізації роботи з записами(пошук, збереження , видалення)
LinguisticChainServiceImpl	Клас	Services.impl	Базова реалізація інтерфейсу LinguisticChainService для реалізації роботи з файлами і отримання аномалій з вхідних даних.
ResultServiceImpl	Клас	Services.impl	Базова реалізація інтерфейсу ResultService для реалізації роботи з результатами(пошук, збереження , видалення)

Продовження таблиці 2.10

TactServiceImpl	Клас	Services.impl	Базова реалізація інтерфейсу TactService для реалізації роботи з тактами(пошук, збереження , видалення)
Alphabet	Перелік	Utils	Перелік ,який використовується для зберігання алфавіту.
LinguisticChainBuider	Клас	Utils	Клас для побудови лінгвістичних ланцюгів
CLIRunner	Клас	_____	Клас для запуску всіх сервісів за допомогою Spring
ApiResponse	Клас	Dto.response	Клас для відповіді і відправки статусу для API частини проекту
JwtAuthenticationResponse	Клас	Dto.response	Клас для відповіді і відправки токену для API частини проекту
LoginRequest	Клас	Dto.request	Клас для узагальнення запиту на логін користувача.
SingUpRequest	Клас	Dto.request	Клас для узагальнення запиту на вхід користувача.

Продовження таблиці 2.10

SecurityConfig	Клас	Config	Конфігурація безпеки в програмі
CurrentUser	Інтерфейс	Security	
CustomUserDetailsService	Клас	Security	Клас для пошуку користувача по різним параметрам в базі даних
JwtAuthenticationEntryPoint	Клас	Security	Клас для обробки помилок для неавторизованих користувачів
JwtAuthenticationFilter	Клас	Security	Клас для перевірки користувача.
JwtTokenProvider	Клас	Security	Клас для генерації токенів.
UserPrincipal	Клас	Security	Загальна модель користувача з правами.
ResultController	Клас	Controller	Клас для точки взаємодії з клієнтською частиною застосунку для аналізу ЕКГ
UserController	Клас	Controller	Клас для точки взаємодії з клієнтською частиною застосунку для роботи з користувачами

Продовження таблиці 2.10

AuthController	Клас	Controller	Клас для точки взаємодії з клієнтською частиною для реєстрації користувача в системі.

2.4 Математичне забезпечення.

Для пошуку аномалій було обрано саме метод лінгвістичних ланцюгів тому що він є швидким та призначений для порівняння коротких інтервалів. (На які можна розбити ЕКГ відповідно тому як показано на рисунку 3).

Основною концепцією даного методу є співвідношення числового проміжку до певної літери. Довжина числового проміжку може бути підібрана різною відповідно до розподілу, обраного символьного алфавіту та самого діапазону значень числового ряду.

Приклад відповідного перетворення показаний на рисунку 1.3, де для спрощення відображення було обрано англійський алфавіт та кожній літері відповідав однаковий проміжок.

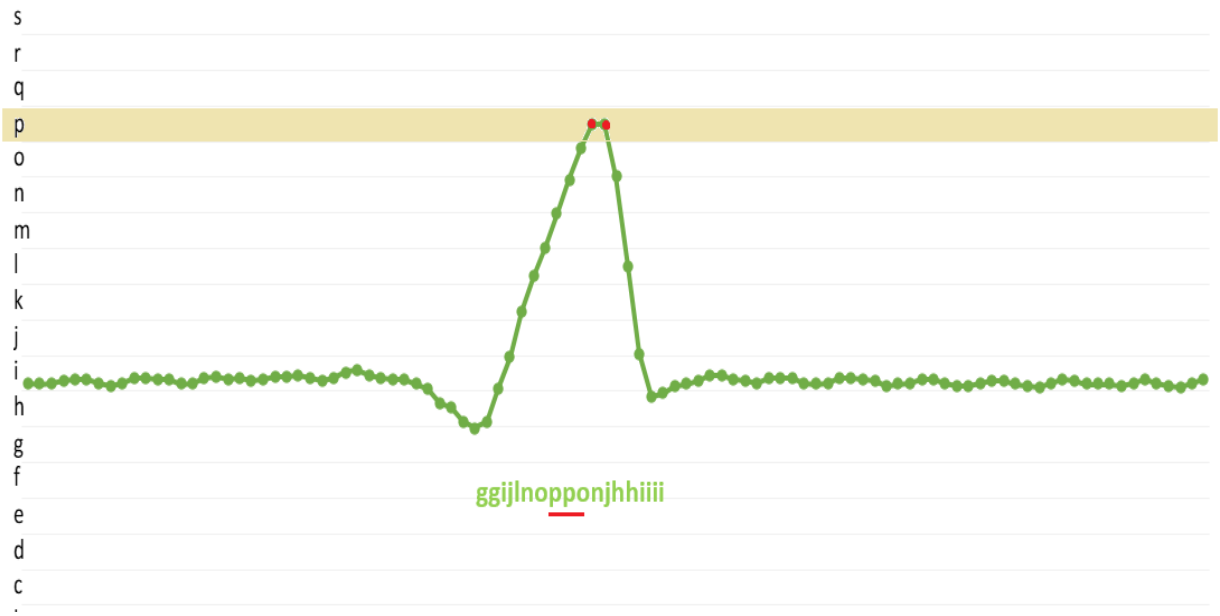


Рисунок 2.5 – Присвоєння числу певної літери

На рисунку 1.3 відображення переведення 2 точок з значенням 1157 ,які відповідають літері р , яка має діапазон значень 1145 по 1168.

Таким чином опрацьовуємо всі вхідні дані.

При використанні лінгвістичного підходу варто зазначити необхідність отримання початкового промаркованого набору з аномаліями для їх пошуку в подальшому. При пошуку аномалій в справжніх даних будуть використані алгоритми для пошуку відстані між рядками :відстань Левеншейна ,відстань Геммінга, Джаро-Вінклера , Дамерау-Левенштейна.

Всі метрики мають в основі три основні властивості , які в свою чергу є у всіх перелічених вище відстанях. Тому в основі 3 з 4 метрик(окрім Джаро-Вінклера) лежать наступні властивості: [30]

- аксіома тотожності – якщо $x=y$ то $d(x,y) = 0;$
- аксіома симетрії – тобто $d(x,y)=d(y,x);$
- аксіома трикутника або нерівність трикутника – $d(x,z)<d(x,y)+d(y,z)$.

Розглянемо детальніше кожен алгоритм вимірювання відстані між символьними ланцюгами.

Алгоритм вимірювання відстані Левенштейна, який є особливим типом вимірювання відстані редагування. Він визначає набір операцій редагування, таких як вставка заміна або видалення терміну, а також вартість кожної операції. Відстань між двома запитами визначається як сума витрат у найдешевшому ланцюжку операцій редагування, що перетворює один рядок в інший. Наприклад відстань між словами «слово» та «олово» є 1 так як потребує заміни одного символу.

Математично відстань Левенштейна для двох символьних рядів a та b з довжиною рядків n та m відповідно можна порахувати за допомогою формули 1.1

$$d(a,b) = d(n,m) \quad (1.1)$$

$$D(i,j) = \begin{cases} 0, i = 0, j = 0 \\ i, j = 0, i > 0 \\ j, j > 0, i = 0 \\ \min \begin{cases} D(i,(j-1)) + 1 \\ D(i-1,j) + 1 \\ D(i-1,j-1) + m(a[i], b[j]) \end{cases} & i > 0, j > 0 \\ \} \end{cases}$$

Де $m(a[i], b[j])$ повертає 0, якщо символи однакові і 1 в іншому випадку.

Алгоритм відстані Геммінга використовується в біоінформатиці та геноміці для аналізу нуклеїнових кислот (ДНК і РНК) та ще багатьох сферах . Вперше проблема підрахунку відстані Геммінга була поставлена Мінський і Паперт в 1969 році [31], де завдання зводилася до пошуку всіх рядків з бази даних, які знаходяться в межах заданої відстані Геммінга до запитуваної.

Відстань Геммінга вже досить широко використовується для різних завдань, таких як пошук близьких дублікатів, розпізнавання образів, класифікація документів, виправлення помилок, виявлення вірусів і т.д.

Наприклад, Манку і співавтори запропонували вирішення проблеми кластеризації дублікатів при індексації веб документів на основі підрахунку відстані Геммінга [32].

Також Міллер і його команда запропонували концепцію пошуку пісень за заданим аудіо фрагменту [33], [34].

Подібні рішення були використані і для завдання пошуку зображень і розпізнавання сітківки [35], [36] і т.д.

Загалом відстань Геммінга становить кількість замін для порівняння рядків. Наведемо приклади:

HD(**10**0, **0**0**1**) = 2.

HD(**со**ва, **ла**ва) = 2

Відстань Дамерау-Левенштейн - розширення на відстань Левенштейна. Він також визначається як мінімальна кількість простих операцій редагування рядка

для зміни його на іншу, але список дозволених операцій розширений. Відстань Дамерау-Левенштейн відрізняється від класичної відстані Левенштейна тим, що включає транспозиції до своїх допустимих операцій на додаток до трьох класичних операцій редагування одного символу (вставки, вилучення та заміни). Даний метод часто використовується для перевірки граматичних помилок при наборі тексту.

Наведемо символьний приклад:

DLD(**10**1,**01**1) = 1

Алгоритм відстані Джаро-Вінклера – модифікований алгоритм знаходження відстані Джаро(1898) представлений світу в 1999 році Вільямом Вінклером.[37][38]

Чим менше відстань Джаро-Вінклера для двох рядків, тим більше схожості мають ці рядки один з одним. Результат нормується, так що відстань рівна 0 означає відсутність схожості, а відстань рівна 1 - точний збіг. Подібність Джаро-Вінклера рівна 1 відняти отриману відстань.

Для кращого розуміння відстані Джаро-Вінклера опишемо відстань Джаро: Загальна формула відстані між двома рядками s_1 та s_2

$$d = \begin{cases} 0 & ,m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & \end{cases} \quad (1.2)$$

Де $|s|$ - довжина рядка

m – число однакових символів , де однаковими символами вважаються однакові символи які знаходяться на відстані не більше z один від одного. (Відстань z описана в формулі 1.3)

t – половина числа транспозицій

$$z = \frac{\max(|s_1|, |s_2|)}{2} - 1 \quad (1.3)$$

Алгоритм вимірювання відстані Джаро-Вінклера в свою чергу використовує коефіцієнт масштабування p , що дає кращі результати в схожих з початку і до певної довжини l , котра називається префіксом.

Загальна формула для відстані Джаро-Вінклера наведена в формулі 1.4.

$$d_w = d_j + (lp(1 - d_j)) \quad (1.4)$$

Де d_j – відстань Джаро

l – довжина префіксу

p – постійний коефіцієнт масштабування

Наведемо приклад:

Порівняємо слова проект та професія

Таблиця 2.11 – Приклад вимірювання відстані методом Джаро-Вінклера.

	П	Р	О	Е	К	Т
П	1	0	0	0		
Р	0	1	0	0	0	
О	0	0	1	0	0	0
Ф	0	0	0	0	0	0
Е		0	0	1	0	0
С			0	0	0	0
І				0	0	0
Я					0	0

Тут максимальна відстань дорівнює 8/2 -1 =3 тому зафарбовані клітинки в яких будуть перевірятись збіг даних.

В наведеному прикладі

$m = 4$

$|s_1| = 6$

$|s_2| = 8$

t =0

Порахуємо відстань Джаро :

$$d_j = \frac{1}{3}\left(\frac{4}{6} + \frac{4}{8} + \frac{4}{4}\right) = 0{,}7(2) \quad (1.5)$$

Очевидно що довжина префіксу рівна 3 і відстань Джаро-Вінклера має наступні значення

$$d_w = 0{,}7(2) + \left(3 * 0{,}1\left(1 - 0{,}7(2)\right)\right) = 0{,}804 \quad (1.6)$$

2.5 Аналіз безпеки даних.

Для надійного зберігання паролей в базі даних потрібно використовувати SHA3[39] для зжимання вхідних даних , таким чином ми можемо використовувати паролі будь якого розміру навіть до декількох гігабайт. Після зжимання за допомогою SHA3 можемо використати Argon2 для шифрування паролей.

Даний алгоритм є стійким до брутфорсу , кольорових таблиць , різних словників, патернів та різних гібридних підходів. Argon2[40] є ключовою функцією виведення, яка була обраний в якості переможця конкурс хешировання пароля в липні 2015. Він був розроблений Алексом Бірюков, Даніель Діном, і Дмитро Ховратовічем з Університету Люксембургу. Еталонна реалізація Argon2 випущена під Creative Commons cc0 ліцензії (тобто суспільного надбання) або Apache License 2.0, і надає три взаємопов'язані версії:

Argon2d збільшує стійкість до атак крекінгу GPU. Він отримує доступ до пам'яті в масиві пароля залежного порядку, що зменшує можливість тимчасової пам'яті компромісних атак, але вводить можливі атаки з бічним каналом.

Argon2i оптимізований, щоб протистояти атакам бокового каналу. Він звертається масив пам'яті в пароль незалежного замовлення.

Argon2id це гібридна версія. Звідси випливає підхід Argon2i для першого проходу над пам'яттю і підхід Argon2d для подальших проходів. Проект

Інтернету рекомендує використовувати Argon2id крім випадків, коли є підстави вважати за краще один з двох інших режимів.

2.6 Висновки по розділу.

В даному розділі був проаналізований структурний підхід до побудови програми на мові програмування Java. За основу була обрана модель MVC завдяки тому що вона дозволяє розподілити обов'язки системи і зробити код більш зрозумілим , також кожна частина стає менш залежною одна від одної.

Описано обрані фреймворки для даної програми і основну мету їх використання.

Продемонстровано роботу основних алгоритмів для пошуку відстані між символьними ланцюгами, а саме Левеншейна, відстань Геммінга, Джаро-Вінклера, Дамерау-Левенштейна. Згодом всі відстані будуть нормалізовані (приведені в значення від 0 до 1) і результуюче значення буде обраховане як середнє арифметичне між цими величинами.

Також проаналізовані і зібрані основні об'єкти для роботи застосунку і описані в даному розділі. Основними серед них являються користувач , аномалія в ЕКГ та результат роботи програми.

Окрім того описане математичне підґрунтя для даної роботи і обрані найкращий алфавіт та алгоритми відстані для отримання найбільш точних результатів. І описані основні технології для забезпечення безпеки даних.

3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Опис процесів тестування

Однією з важливих фаз є тестування отриманого продукту.

– Функціональність - "Сукупність атрибутів, що залежать від існування набору функцій та їх визначених властивостей. Функції - це ті, які задовольняють заявлені або маються на увазі.

– Надійність - "Сукупність атрибутів, які залежать від можливостей програмного забезпечення підтримувати рівень продуктивності за заявлених умов протягом зазначеного періоду часу".

– Корисність - "Набір атрибутів, що покладаються на зусилля, необхідні для використання, та на індивідуальну оцінку такого використання заявленим або мається на увазі набором.

– Ефективність - "Сукупність атрибутів, які залежать від рівня продуктивності програмного забезпечення та кількості використовуваних ресурсів у зазначених умовах".

– Ремонтопридатність - "Набір атрибутів, які докладають зусиль, необхідних для внесення .

– Відповідність технічному обслуговуванню.

– Переносимість - "Набір атрибутів, що залежать від здатності програмного забезпечення переноситися з одного середовища в інше".

3.2 Опис сценаріїв тестування

Основні сценарії які мають бути протестовані.

– Реєстрація користувача з валідними даними.

– Реєстрація користувача з поштовою скринькою.

– Додавання додаткових даних для користувача.

– Реєстрація користувача з логіном ,який вже є в системі.
– Додавання пошти користувача.
– Авторизація користувача в системі.
– Авторизація користувача в системі з неправильним паролем.
– Відправлення файлу з ЕКГ для подальшого аналізу.
– Отримання результатів аналізу ЕКГ поштою.

3.3 Сценарії тестування

Таблиця 3.1 – Тест кейс з реєстрацією нового користувача

Назва тесту	Реєстрація нового користувача
Мета тесту	Перевірка функціоналі щодо створення нового користувача в системі.
Критичність для системи	Дуже висока
Початкова URL адреса	*/api/auth/signup*
HTTP метод	POST

Тіло запиту	{
	"username": "",
	"password": ""
	}

Початковий стан	Відсутні користувачі з введеним логіном
Вхідні дані	Логін користувача – рядок не коротше 3 символів, який не містить спеціальних позначень Пароль користувача – рядок не менше 6 символів
Сценарій проведення тесту	Заповнити відповідні дані в запиті та надіслати його
Очікуваний результат	Створення нового користувача та додавання відповідної інформації в базу даних.

Продовження таблиці 3.1

Зміни в системі після виконання даного	Створення нового користувача та додавання відповідної інформації в базу даних . Створення унікального ідентифікатора для облікового запису.

Таблиця 3.2 – Тест кейс з реєстрацією нового користувача з поштою

Назва тесту	Реєстрацією нового користувача з поштовою скринькою
Мета тесту	Перевірка функціоналі щодо створення нового користувача в системі з поштовою скринькою.
Критичність для системи	Висока.
Початкова URL адреса	*/api/auth/signup*
HTTP метод	POST

Тіло запиту	{
	"username": "",
	"password": ""
	"email": ""
	}

Початковий стан	Відсутні користувачі з введеним логіном та поштовою скринькою.
Вхідні дані	Логін користувача – рядок не коротше 3 символів, який не містить спеціальних позначень Пароль користувача – рядок не менше 6 символів Поштова скринька – містить @ та .com
Сценарій проведення тесту	Заповнити відповідні дані в запиті та надіслати його
Очікуваний результат	Створення нового користувача та додавання відповідної інформації в базу даних. Лист для перевірки поштової скриньки відправлений.

Продовження таблиці 3.2

Зміни в системі після виконання даного	Створення нового користувача та додавання відповідної інформації в базу даних . Створення унікального ідентифікатора для облікового запису. Відправка листа на поштову скриньку.

Таблиця 3.3 – Тест кейс з реєстрацією нового користувача з логіном який є в системі

Назва тесту	Реєстрацією нового користувача з логіном ,який існує в системі
Мета тесту	Перевірка функціоналі щодо створення нового користувача в системі , якщо відповідний логін вже зайнятий.
Критичність для системи	Дуже висока.
Початкова URL адреса	*/api/auth/signup*
HTTP метод	POST

Тіло запиту	{
	"username": "",
	"password": ""
	}

Початковий стан	Існує користувач з введеним логіном
Вхідні дані	Логін користувача – рядок не коротше 3 символів, який не містить спеціальних позначень Пароль користувача – рядок не менше 6 символів Поштова скринька – містить @ та .com
Сценарій проведення тесту	Заповнити відповідні дані в запиті та надіслати його
Очікуваний результат	Новий користувач не створений і відповідна помилка про існування даного користувача передана.

Продовження таблиці 3.3

Зміни в системі після виконання даного	Новий користувач не створений і відповідна помилка про існування даного користувача передана..

Таблиця 3.4 – Тест кейс з додавання додаткових даних для користувача

Назва тесту	Додання нових даних про користувача
Мета тесту	Перевірка функціоналі щодо додання нових даних до облікового запису користувача , який вже існує в системі.
Критичність для системи	Низька.
Початкова URL адреса	*/api/auth/singin*
HTTP метод	POST

Тіло запиту	{
	"weight": "",
	"height": "",
	"age": ""
	"sex": ""
	}

Початковий стан	Користувач існує та додаткові данні про нього не надані. Відповідний користувач проводить дії над своїм обліковим записом.
Вхідні дані	Поля ріст, вага , вік , стать є цілочисельними Кожен з них не є обов’язковим але повинен бути заповнений хоча б один параметр
Сценарій проведення тесту	Заповнити відповідні дані в запиті та надіслати його
Очікуваний результат	Додаткові дані додані до користувача і відображаються на його обліковому записі.

Продовження таблиці 3.4

Зміни в системі після виконання даного	Додаткові дані додані до користувача і відображаються на його обліковому записі. Також дані збережені в базі даних.

Таблиця 3.5 – Тест кейс з авторизацією користувача в системі

Назва тесту	Авторизація користувача в системі
Мета тесту	Перевірка функціоналі щодо можливості реєстрації користувача в системі
Критичність для системи	Дуже висока.
Початкова URL адреса	*api/auth/singin*
HTTP метод	POST

Тіло запиту	{
	"username": "",
	"password": ""
	}

Початковий стан	Користувач з даним логіном та паролем існує.
Вхідні дані	Логін користувача – рядок не коротше 3 символів, який не містить спеціальних позначень Пароль користувача – рядок не менше 6 символів
Сценарій проведення тесту	Заповнити відповідні дані в запиті та надіслати його
Очікуваний результат	Користувач успішно увійшов в систему.
Зміни в системі після виконання даного	Користувач успішно увійшов в систему.

Таблиця 3.6 – Тест кейс з авторизацією користувача в системі

Назва тесту	Авторизація користувача в системі
Мета тесту	Перевірка функціоналі щодо перевірки паролю при авторизації користувача в системі.
Критичність для системи	Дуже висока.

Продовження таблиці 3.6

Початкова URL адреса	*api/auth/singin*
HTTP метод	POST

Тіло запиту	{
	"username": "",
	"password": ""
	}

Початковий стан	Користувач з даним логіном існує але пароль відрізняється.
Вхідні дані	Логін користувача – рядок не коротше 3 символів, який не містить спеціальних позначень Пароль користувача – рядок не менше 6 символів
Сценарій проведення тесту	Заповнити відповідні дані в запиті та надіслати його
Очікуваний результат	Користувач не увійшов в систему і була відправлена помилка про некоректно введені данні.
Зміни в системі після виконання даного	Користувач не увійшов в систему і була відправлена помилка про некоректно введені данні.

Таблиця 3.7 – Тест кейс з відправлення даних ЕКГ

Назва тесту	Відправлення даних ЕКГ для подальшого аналізу
Мета тесту	Перевірка функціоналі щодо відправки файлу з даними ЕКГ на подальший аналіз
Критичність для системи	Дуже висока.
Початкова URL адреса	*api/ekg/analytic*
HTTP метод	POST

Продовження таблиці 3.7

Тіло запиту	{
	"file": "",
	}

Початковий стан	Дана операція є доступною для будь-якого користувача.
Вхідні дані	Файл з даними ЕКГ
Сценарій проведення тесту	Заповнити відповідні дані в запиті та надіслати його
Очікуваний результат	Дані ЕКГ були проаналізовані і був відправлений результат.
Зміни в системі після виконання даного	Дані ЕКГ були проаналізовані і був відправлений результат. Знайдені аномалії були записані в відповідну таблицю в базі даних.

Таблиця 3.8 – Тест кейс з відправлення результату ЕКГ на пошту

Назва тесту	Відправлення даних ЕКГ для подальшого аналізу з відправкою на пошту
Мета тесту	Перевірка функціоналі щодо відправки файлу з результатами аналізу ЕКГ на пошту.
Критичність для системи	Дуже висока.
Початкова URL адреса	*api/ekg/analytic?post=true*
HTTP метод	POST

Тіло запиту	{
	"file": "",
	}

Початковий стан	Користувач авторизований в системі з вказаною поштою
Вхідні дані	Файл з даними ЕКГ

Продовження таблиці 3.8

Сценарій проведення тесту	Заповнити відповідні дані в запиті та надіслати його
Очікуваний результат	Дані ЕКГ були проаналізовані і був відправлений результат на пошту.
Зміни в системі після виконання даного	Дані ЕКГ були проаналізовані і був відправлений результат на пошту. Знайдені аномалії були записані в відповідну таблицю в базі даних.

4 ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Розгортання програмного забезпечення

Дана програма може бути запущена з будь-якого типу пристрою – телефону, планшету або комп'ютеру завдяки тому що воно є браузерним. Мінімальна конфігурація пристроїв:

Процесор - Intel Core i3+.

Об'єм ОЗП – 1ГБ

Об'єм фізичної пам'яті: 64Гб

Даний веб-застосунок може запускатися в Chrome, Safari та Opera веб браузерах

4.2 Публікація програмного забезпечення

Для того щоб опублікувати серверну частини з базою даних необхідно виконати наступні кроки:

– встановити JDK 8 версії або вище з офіційного сайту https://www.oracle.com/java/technologies/javase-downloads.html;

– встановити PostgreSQL базу даних з офіційного сайту. https://www.postgresql.org/ ;

– встановити Maven 3.5 версії або вище з офіційного сайту http://maven.apache.org/download.cgi і згодом перевірте версію за допомогою консольної команди mvn -version;

– завантажити відповідний проект за посиланням https://github.com/AnnaTsts/EckAnalytics ;

– відкрити кореневу папку проекту;

– виконати команда mvn clean install;

– виконати команду java -jar backend-0.01-SNAPSHOT.jar .

4.3 Висновок до розділу

В даному розділі наведена основна інформація про використання даного продукту та інструкція для розгортання програми.

ВИСНОВКИ

Даний дипломний проєкт присвячений аналізу роботи м’яза міокарда шляхом обробки та пошуку аномалій в ЕКГ. Було виявлено достатньо багато аналогів в даній області, але більшість з них потребують великої кількості маркованих даних та абсолютно всі оперують числами для аналізу даних.

В якості математичного апарату було взято лінгвістичний моделювання, а на його основі запропоновано новий підхід до аналізу ЕКГ та виявлення аномалій – використання лінгвістичних ланцюгів та символьних відстаней для приблизного пошуку. Було використані найбільш поширені алгоритми для пошуку відстані такі як Геммінга, Джаро-Вінклера , Дамерау-Левенштейна ,які допомагають визначити відстань за допомогою середнього арифметичного між ними. Завдяки простим операціям даний метод виявився дуже швидким для такого великого об’єму інформації.

Отриманий програмний продукт складається з двох компонентів – серверної частини для обробки і зберігання даних і клієнтської частини для надання зручного інтерфейсу. Було створено серверну частину за допомогою мови програмування Java з використанням передових технологій для даної мови таких як Spring та Hibernate. Для клієнтської частини було використано Angular та його основні компоненти. Даний додаток надає функціонал реєстрації та авторизації користувачів , обробки ЕКГ.

В ході розробки були продумані основні сценарії використання даного продукту, проведено тестування найкращого підходу до використання лінгвістичного алгоритму.

Після написання проекту було проведено тести основних сценаріїв та розроблена детальна інструкція по експлуатації даного продукту.

Розроблений веб-застосунок виконує основні свої функції. Мета проєкту досягнуто за рахунок реалізації лінгвістичного методу для аналізу електрокардіограм.

ПЕРЕЛІК ПОСИЛАНЬ

1) Sanders W. B. Heart Disease. A textbook of Cardiovascular Medicine. / Sanders. // 1. – 1980. – С. 1427.

2) Клиническая информатика и Телемедицина. // 9. – 2013. – №10. – С. 48–56.

3) Баклан І. В. Лінгвістичне моделювання: основи, методи, деякі прикладні аспекти / І. В. Баклан // Систем. технології. — 2011. — № 3. — С. 10-19.

4) W. Chang and J. Lampe. Theoretical and empirical comparisons of approximate string matching algorithms. In Proc. 3rd Combinatorial Pattern Matching (CPM'92), LNCS 644, pages 172{181, 1992.

5) P. Sellers, The theory and computation of evolutionary distances: Pattern recognition, J. Algorithms 1 (1980) 359–373.

6) Baklan, I., Mukha, I., Oliinyk, Y., Lishchuk, K., Nedashkivsky, E., & Gavrilenko, O. (2019, January). Anomalies Detection Approach in Electrocardiogram Analysis Using Linguistic Modeling. In International Conference on Computer Science, Engineering and Education Applications (pp. 513-522). Springer, Cham.

7) Yoo, P. D. (2019). An Enhanced Machine Learning Based Biometric Authentication System Using RR-Interval Framed Electrocardiograms. *arXiv preprint arXiv:1907.13517*.

8) Ding, Z., Qiu, S., Guo, Y., Lin, J., Sun, L., Fu, D., ... & Lv, T. (2019). LabelECG: A Web-based Tool for Distributed Electrocardiogram Annotation. arXiv preprint arXiv:1908.06553.

9) Das, A. K., Catthoor, F., & Schaafsma, S. (2018, September). Heartbeat Classification in Wearables Using Multi-layer Perceptron and Time-Frequency Joint Distribution of ECG. In 2018 IEEE/ACM International Conference on Connected Health: Applica-tions, Systems and Engineering Technologies (CHASE) (pp. 69-74). IEEE.\

10) Sun, H., Ganglberger, W., Panneerselvam, E., Leone, M. J., Quadri, S. A. Q., Goparaju, B., ... & Westover, M. B. (2019). Sleep Staging from Electrocardiography and Respira-tion with Deep Learning. arXiv preprint arXiv:1908.11463.

11) Arsene, C. (2019). Complex Deep Learning Models for Denoising of Human Heart ECG signals. arXiv preprint arXiv:1908.10417.

12) Liu, Y., He, R., Wang, K., Li, Q., Sun, Q., Zhao, N., & Zhang, H. (2019). Automatic De-tection of ECG Abnormalities by using an Ensemble of Deep Residual Networks with Attention. arXiv preprint arXiv:1908.10088.

13) Yuan, B., & Xing, W. (2019). Diagnosing Cardiac Abnormalities from 12-Lead Elec-trocardiograms Using Enhanced Deep Convolutional Neural Networks. arXiv pre-print arXiv:1908.06802.

14) Hsu, C. C. Y., Hardt, M., & Hardt, M. (2019). Linear Dynamics: Clustering without identification. arXiv preprint arXiv:1908.01039.

15) Contoyiannis, Y., Diakonos, F., & Kampitakis, M. (2019). Applying the Method of Critical Fluctuations on Human Electrocardiograms. arXiv preprint arXiv:1908.06408.

16) Wang, S. C., Wu, H. T., Huang, P. H., Chang, C. H., Ting, C. K., & Lin, Y. T. (2019). Novel imaging revealing inner dynamics for cardiovascular waveform analysis via un-supervised manifold learning. arXiv preprint arXiv:1909.04206.

17) Kachhara, S., & Ambika, G. (2019). Bimodality and Scaling in Recurrence Networks from ECG data. arXiv preprint arXiv:1908.01286.

18) Learn Objective-C for Java Developers. – 353-402 с.

19) Majeed A. MVC Architecture: A Detailed Insight to the Modern Web Applications Development / Abdul Majeed. – 2018.

20) Дашнер С. Изучаем Java EE. Современное программирование для больших предприятий / Себастьян Дашнер. – Питер, 2018.

21) Walls C. Spring in Action / Craig Walls.. – (Fifth Edition).

22) Advantages of Spring Framework With Limitations [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://data-flair.training/blogs/advantages-of-spring/#:~:text=One%20of%20the%20major%20benefits,in%20order%20to%20develop%20applications

23) hibernate [Електронний ресурс] – Режим доступу до ресурсу: https://hibernate.org/.

24) Kirpalsinh R. Basics of Hibernate: Easy Learning Hibernate / Raj Kirpalsinh., 2013. – (9-13).

25) Hibernate Community Documentation [Електронний ресурс]. – 2004. – Режим доступу до ресурсу: https://docs.jboss.org/hibernate/core/3.3/reference/en/html/transactions.html.

26) Hibernate - Query Language [Електронний ресурс] – Режим доступу до ресурсу: https://www.tutorialspoint.com/hibernate/hibernate_query_language.htm.

27) Hibernate - Configuration [Електронний ресурс] – Режим доступу до ресурсу: https://www.tutorialspoint.com/hibernate/hibernate_configuration.htm.

28) Angular [Електронний ресурс] – Режим доступу до ресурсу: https://angular.io/.

29) postgresql [Електронний ресурс] – Режим доступу до ресурсу: https://www.postgresql.org/.

30) Распознавание образов и анализ сцен. – 367 с.

31) Hamming, R. W. (April 1950). "Error detecting and error correcting codes" (PDF). The Bell System Technical Journal. 29 (2): 147–160. doi:10.1002/j.1538-7305.1950.tb00463.x. ISSN 0005-8580.

32) Влади́мир И. Левенштейн (1965). Двоичные коды с исправлением выпадений, вставок и замещений символов [Binary codes capable of correcting deletions, inser-tions, and reversals]. Доклады Академий Наук СССР (in Russian). 163 (4): 845–8. Appeared in English as: Levenshtein, Vladimir I. (February 1966). "Binary codes ca-pable of correcting deletions, insertions, and reversals". Soviet Physics Doklady. 10 (8): 707–710. Bibcode:1966SPhD...10..707L.

33) Jaro, M. A. (1989). "Advances in record linkage methodology as applied to the 1985 census of Tampa Florida". Journal of the American Statistical Association. 84 (406): 414–20. doi:10.1080/01621459.1989.10478785.

34) Levenshtein, Vladimir I. (February 1966), "Binary codes capable of correcting dele-tions, insertions, and reversals", Soviet Physics Doklady, 10 (8): 707–710

35) Damerau, Fred J. (March 1964), "A technique for computer detection and correction of spelling errors", Communications of the ACM, 7 (3): 171–176, doi:10.1145/363958.363994

36) The method used in: Majorek, Karolina A.; Dunin-Horkawicz, Stanisław; et al. (2013), "The RNase H-like superfamily: new members, comparative structural analy-sis and evolutionary classification", Nucleic Acids Research, 42 (7): 4160–4179, doi:10.1093/nar/gkt1414, PMC 3985635, PMID 24464998

37) Jaro, M. A. Probabilistic linkage of large public health data file // Statistics in Medicine : journal. — 1995. — Vol. 14, no. 5—7. — P. 491—498.

38) William E. Winkler. Overview of Record Linkage and Current Research Directions // Research Report Series, RRS : journal. — 2006.

39) SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions | NIST

40) Joel Alwen, Jeremiah Blocki. Efficiently Computing Data-Independent Memory-Hard Functions. — Advances in Cryptology – CRYPTO 2016, 2016. — P. 241—271

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

“ЗАТВЕРДЖЕНО”

В.о. завідувача кафедри

_____ Олександр ПАВЛОВ

“___” _____ 2020 р.

Веб-застосування для виявлення аномалій в електрокардіограмах лінгвістичним методом

Технічне завдання

КПІ.ІП-6125.045420.02.91

“ПОГОДЖЕНО”

Керівник проєкту:

_____ Ю.О Олійник

Нормоконтроль:

_____ К.І. Ліщук

Виконавець:

_____ А.В Цицилюк

Київ – 2020 року

ЗМІСТ

1 НАЙМЕНУВАННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ

Назва розробки: Веб-застосування для виявлення аномалій в електрокардіограх лінгвістичним методом

Галузь застосування: Мережа Інтернет та галузь охорони здоров’я**.**

Наведене технічне завдання поширюється на розробку Веб-застосування аналізу завантажених даних ЕКГ та надання прогнозу про можливі аномалії в даному ЕКГ.

2 ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки Веб-застосування для виявлення аномалій в електрокардіограмах лінгвістичним методом на дипломне проектування, затверджене кафедрою автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» («КПІ ім. Ігоря Сікорського»).

3 ПРИЗНАЧЕННЯ РОЗРОБКИ

Основним призначення даного програмного застосунку є своєчасне виявлення аномалій в роботі серця задля усунення подальших хвороб або критичних станів серця.

Загалом даний продукт буде корисний лікарям для локалізації аномалій та для пацієнтів для первинного аналізу ЕКГ.

Також метою даної роботи є вивчення корисності лінгвістичного методу в сфері аналізу ЕКГ.

4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Вимоги до функціональних характеристик

4.1.1 Програмне забезпечення повинно забезпечувати виконання наступних основних функції:

4.1.1.1 Для користувача:

- авторизація;
- реєстрація;
- перегляд облікових даних;
- редагування облікових даних;
- завантаження файлу з ЕКГ;
- запит на проведення пошуку аномалій в ЕКГ.

4.1.2 Розробка проводилась на платформі MAC OS, але серверну частину веб застосунку можна розгортати на будь-якій платформі

4.1.3 Додаткові вимоги.

Не передбачені.

4.2 Вимоги до надійності

4.2.1 Передбачити контроль введення інформації

4.2.2 Передбачити захист від некоректних дій користувача

4.2.3 Передбачити уникнення втрат персональних даних користувачів

4.2.4 Забезпечити цілісність інформації в базі даних

4.3 Умови експлуатації

4.3.1 Умови експлуатації згідно СанПін 2.2.2.542 – 96

Не висуваються

4.3.2 Обслуговування

Окрім встановлення та запуску серверної частини додатку обслуговування не передбачається.

4.4 Вимоги до складу і параметрів технічних засобів

4.4.1 Програмне забезпечення повинно функціонувати на IBM-сумісних персональних комп‘ютерах

4.4.2 Мінімальна конфігурація технічних засобів

4.4.2.1 Тип процесор

64-розрядний процесор з тактовою частотою не нижче 1.4 ГГц

4.4.2.2 Об‘єм ОЗП

Не менше 1 ГБ

4.4.2.3 Об’єм жорсткого диску

Не менше 10 ГБ.

4.5 Вимоги до інформаційної та програмної сумісності

Оскільки даний застосунок є веб застосунком – особливі умови до користувачів не висуваються.

4.6 Вимоги до маркування та пакування

Вимоги до маркування та пакування не висуваються.

4.7 Вимоги до транспортування та зберігання

Вимоги до транспортування та зберігання не висуваються.

4.8 Спеціальні вимоги

Має бути створена установча версію програмного забезпечення.

5 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

5.1 Попередній склад програмної документації:

а) Супроводжувальна документація:

1) Пояснювальна записка;

2) Технічне завдання;

3) Керівництво користувача.

б) Довідникова документація:

1) Програмні модулі, котрі розробляються, повинні бути задукоментовані, тобто тексти програм повинні містити всі необхідні коментарі.

в) Графічна документація:

1) Схема структури бази даних;

2) Креслення екранних форм;

3) Схема структурна класів програмного забезпечення.

6 СТАДІЇ І ЕТАПИ РОЗРОБКИ

№	Назва етапу	Строк,	Звітність
1.	Вивчення літератури за тематикою проекту	28.02.2020	Опис існуючих технічних рішень
2.	Аналіз існуючих методів розв’язання задачі	07.03.2020	Опис аналогів
3.	Постановка та формалізація задачі	12.03. 2020	Технічне завдання
4.	Аналіз вимог до програмного забезпечення	17.03. 2020	Схема структурна програмного забезпечення та специфікація компонентів
5.	Алгоритмізація задачі	25.03. 2020	Додаток з схемою використаного алгоритму
6.	Моделювання програмного забезпечення	29.03. 2020	Технічна документація
7.	Обґрунтування використовуваних технічних засобів	04.04. 2020	Пояснювальна записка.
8.	Розробка архітектури програмного забезпечення	19.04. 2020	Пояснювальна записка.
9.	Розробка програмного забезпечення	02.05. 2020	Тексти програмного забезпечення
10	Налагодження програми	06.05. 2020	Тести, результати тестування
11	Виконання графічних документів	09.05.2020	Графічний матеріал проекту

7 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ

7.1 Види випробувань

а) Тестування розробленого програмного продукту виконується відповідно до 4 пункту Пояснювальної записки.

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

“ЗАТВЕРДЖЕНО”

В.о. завідувача кафедри

_____ Олександр ПАВЛОВ

“___” _____ 2020 р.

Веб-застосування для виявлення аномалій в електрокардіограмах лінгвістичним методом

Керівництво користувача

КПІ.ІП-6125.045420.03.34

“ПОГОДЖЕНО”

Керівник проєкту:

_____ Ю.О.Олійник

Нормоконтроль:

_____ К.І. Ліщук

Виконавець:

_____А.В. Цицилюк

Київ – 2020 року

ЗМІСТ

1 ІНСТРУКЦІЯ КОРИСТУВАЧА

1.1 Верхня панель

В верхній частині екрану є основна панель навігації по сайту. Таким чином маємо на ній такі клавіши як реєстрація «Sing Up» , вхід в свій акаунт «Sing In» та сам логотип сайту який є переходом на головну сторінку.

Рисунок 1.1 – Вигляд панелі для неавторизованого користувача

При натисканні кнопки «Sing In» користувач переходить за адресою “sing-in” і може заповнити форму для входу.

Рисунок 1.2 – Виділення кнопки «Sing In» для входу в акаунт

При натисканні кнопки «Sing Up» користувач переходить за адресою “sing-up” і може заповнити форму для реєстрації.

Рисунок 1.3 – Виділення кнопки «Sing Up» для реєстрації

При натисканні символу ЕКГ відбувається перехід на головну сторінку веб додатку.

Рисунок 1.4 – Виділення кнопки для переходу на головний екран

При вході в акаунт користувача верхня панель змінюється і з’являється кнопка з інформацією користувача «Welocome,» і ім’я користувача. Кнопка «Logout» для виходу з акаунту користувача та кнопка переходу на головну сторінку.

Рисунок 1.5 – Вигляд панелі для авторизованого користувача з іменем Anna

Рисунок 1.6 – Кнопка для переходу на акаунта користувача

Рисунок 1.7 – Кнопка для виходу з акаунта користувача

1.2 Реєстрація та авторизація

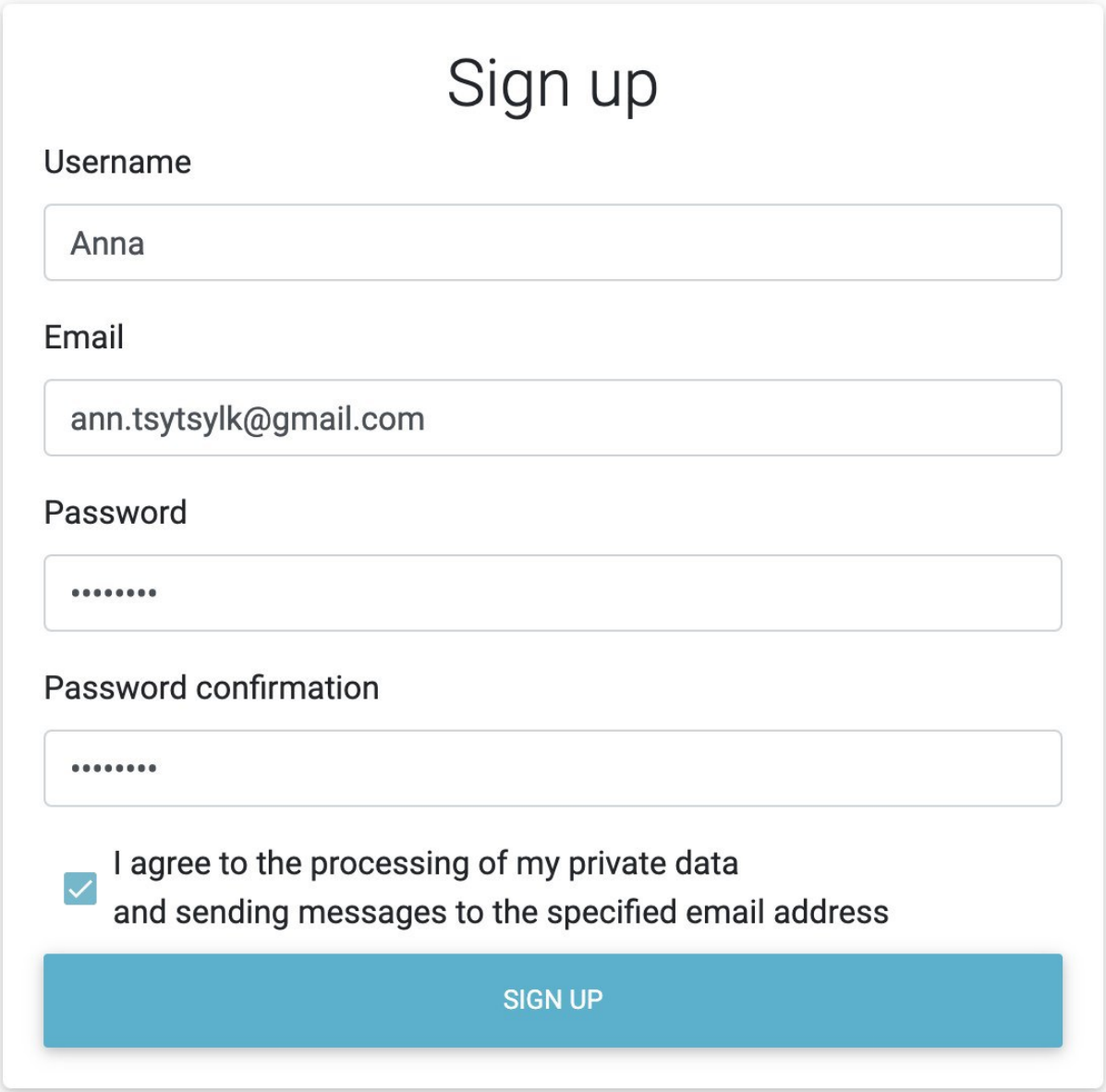


Рисунок 1.8 – Вікно реєстрації

Після натиску на кнопку «Sing Up» з'являється вікно реєстрації користувача. Всі поля повинні бути заповнені для успішної реєстрації.

Поле Username повинно мати більше 3 символів і містити лише літери (Воно відповідає за ім'я користувача в системі)

Поле email повинно відповідати шаблону електронної пошти.(Відповідає за електронну пошту користувача).

Поле Password відповідає за пароль користувача і повинно містити не менше 8 символів.

Повинно також бути відмічено поле з згодою про обробку персональних даних .

Після натиску кнопки реєстрації , якщо всі поля заповнені вірно і немає користувача з таким логіном і паролем система покаже повідомлення про успішну реєстрацію.

Після цього при натиску кнопки «Sing In» , де користувач ввівши своє ім’я в системі та пароль отримують доступ до свого профіля.

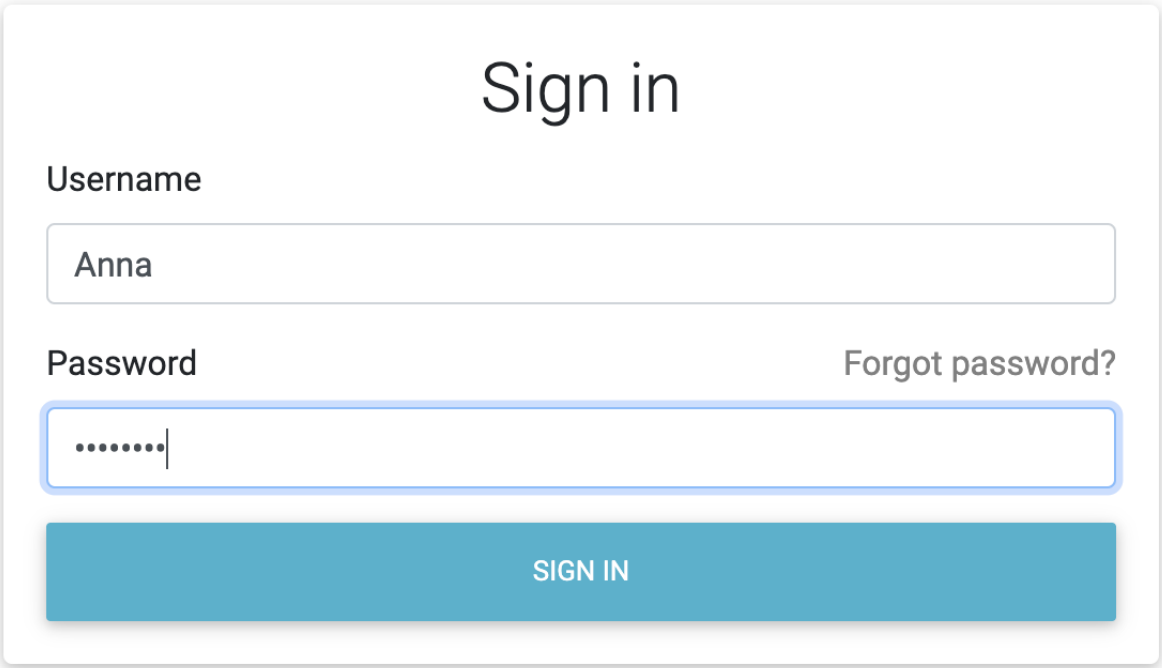


Рисунок 1.9 – Вікно авторизації

1.3 Головний екран

Головний екран носить інформативний характер і стимулює проаналізувати і перевірити свою ЕКГ.

Рисунок 1.10 – Головний екран

За допомогою кнопки на головному екрані «Analyze EKG»

Рисунок 1.11 – Сторінка з аналізом глюкози

При натиску кнопки аналізу потрібно обрати файл з ЕКГ і система проаналізує його і виведе графіки з можливими аномаліями.

Рисунок 1.12 – Графіки з можливими аномаліями.

1.4 Дані користувача.

При переході на сторінку користувача можна побачити його інформацію і потім за допомогою кнопки «Edit» їх редагувати.

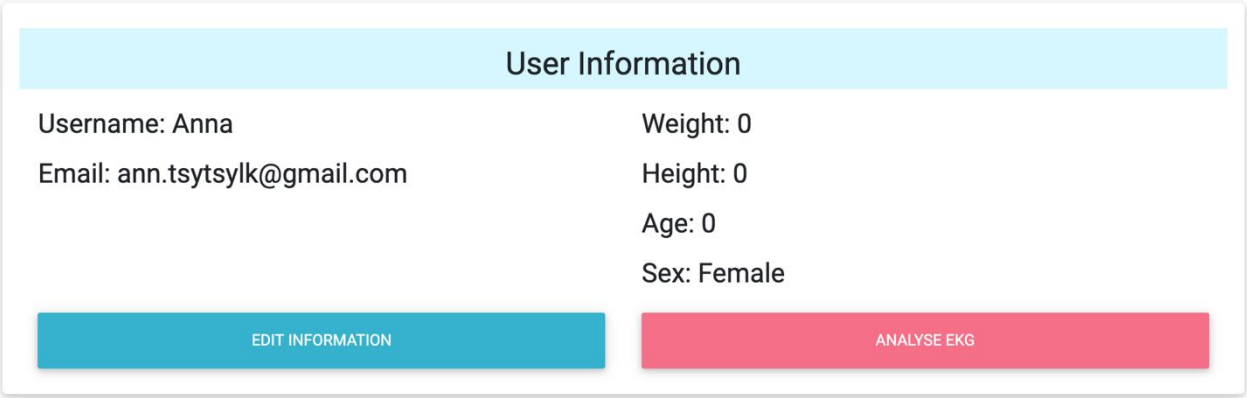


Рисунок 1.13 – Профіль користувача

Рисунок 1.14– Редагування профілю користувача.

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

“ЗАТВЕРДЖЕНО”

В.о. завідувача кафедри

_____ Олександр ПАВЛОВ

“___” _____ 2020 р.

Веб-застосування для виявлення аномалій в електрокардіограмах лінгвістичним методом

Опис програми

КПІ.ІП-6125 045420.04.13

“ПОГОДЖЕНО”

Керівник проєкту:

_____ Ю. О. Олійник

Нормоконтроль:

_____ К.І. Ліщук

Виконавець:

_____А. В. Цицилюк

Київ – 2020 року

Тексти програмного коду

Веб-додаток для виявлення аномалій в електрокардіограмах лінгвістичним методом

(Найменування програми (документа))

DVD-R

(Вид носія даних)

34 арк, 13 Мб

(Обсяг програми (документа) , арк.,) Кб)

Київ – 2020

```
package com.eck_analytics.Algorithm;

public class DamerauLevenshtein {

    /**

     * Calculates the string distance between source and target strings using

     * the Damerau-Levenshtein algorithm. The distance is case-sensitive.

     *

     * @param source The source String.

     * @param target The target String.

     * @return The distance between source and target strings.

     * @throws IllegalArgumentException If either source or target is null.

     */

    public int calculateDistance(CharSequence source, CharSequence target) {

        if (source == null || target == null) {

            throw new IllegalArgumentException("Parameter must not be null");

        }

        int sourceLength = source.length();

        int targetLength = target.length();

        if (sourceLength == 0) return targetLength;

        if (targetLength == 0) return sourceLength;

        int[][] dist = new int[sourceLength + 1][targetLength + 1];

        for (int i = 0; i < sourceLength + 1; i++) {

            dist[i][0] = i;

        }

        for (int j = 0; j < targetLength + 1; j++) {

            dist[0][j] = j;

        }

        for (int i = 1; i < sourceLength + 1; i++) {

            for (int j = 1; j < targetLength + 1; j++) {

                int cost = source.charAt(i - 1) == target.charAt(j - 1) ? 0 : 1;

                dist[i][j] = Math.min(Math.min(dist[i - 1][j] + 1, dist[i][j - 1] + 1), dist[i - 1][j - 1] + cost);

                if (i > 1 &&

                        j > 1 &&

                        source.charAt(i - 1) == target.charAt(j - 2) &&

                        source.charAt(i - 2) == target.charAt(j - 1)) {

                    dist[i][j] = Math.min(dist[i][j], dist[i - 2][j - 2] + cost);

                }

            }

        }

        return dist[sourceLength][targetLength];

    }

}
```

```
package com.eck_analytics.Algorithm;


import org.apache.commons.math3.analysis.interpolation.LinearInterpolator;

import org.apache.commons.math3.analysis.polynomials.PolynomialSplineFunction;


public class XCompression implements Compression {

    @Override

    public char[] release(char[] source, double size) {

        PolynomialSplineFunction interpolate = getPolynomialFunction(source);

        int resultSize = (int) (source.length * size);

        char[] result = new char[resultSize];

        double j = 0;


        for (int i = 0; i < resultSize; i++) {

            result[i] = (char) interpolate.value(j);

            j += size;

        }


        return result;

    }



    private PolynomialSplineFunction getPolynomialFunction(char[] source) {
```

```
        LinearInterpolator linearInterpolator = new
LinearInterpolator();
        double[] x = new double[source.length];
        double[] y = new double[source.length];
        for (int i = 0; i < source.length; i++) {
            double currValue = source[i];
            x[i] = i;
            y[i] = currValue;
        }
        return linearInterpolator.interpolate(x, y);
    }
}

package com.eck_analytics.Algorithm;

public class YCompression implements Compression {
    @Override
    public char[] release(char[] source, double size) {

        char[] result = new char[source.length];
        for (int i = 0; i < source.length; i++) {
            result[i] = (char) (source[i] * size);
        }
        return result;
    }
}
```

```
package com.eck_analytics.Algorithm;


public interface Compression {
    char[] release(char[] source, double size);
}


package com.eck_analytics.Controller;


import
org.springframework.web.bind.annotation.CrossOrigin;
import
org.springframework.web.bind.annotation.RequestMapping;
import
org.springframework.web.bind.annotation.RestController;


@RestController
@RequestMapping("/api/anomaly")
@CrossOrigin("http://localhost:4200")
public class AnomalyController {
}


package com.eck_analytics.Controller;


import com.eck_analytics.Services.impl.AuthService;
import com.eck_analytics.dto.request.LoginRequest;
import com.eck_analytics.dto.request.SignUpRequest;
import
com.eck_analytics.dto.response.JwtAuthenticationResponse;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import
org.springframework.security.authentication.Authentication
Manager;
import
org.springframework.security.crypto.password.PasswordEnc
oder;
import org.springframework.web.bind.annotation.*;
import com.eck_analytics.security.JwtTokenProvider;



import javax.validation.Valid;
@RestController
@RequestMapping("/api/auth")
@CrossOrigin("http://localhost:4200")
public class AuthController {
    @Autowired
    private AuthenticationManager authenticationManager;


    @Autowired
    private PasswordEncoder passwordEncoder;


    @Autowired
    private JwtTokenProvider tokenProvider;


    @Autowired
    private AuthService authService;


    @PostMapping("/signin")
    public ResponseEntity<?> authenticateUser(@Valid
@RequestBody LoginRequest loginRequest) {
        return ResponseEntity.ok(new
JwtAuthenticationResponse(authService.authenticateUser(
                loginRequest, authenticationManager,
tokenProvider)));
    }


    @PostMapping("/signup")
    public ResponseEntity<?> registerUser(@Valid
@RequestBody SignUpRequest signUpRequest) {
        return authService.registerUser(signUpRequest,
passwordEncoder);
    }


}


package com.eck_analytics.Controller;


import lombok.extern.slf4j.Slf4j;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.multipart.MultipartFile;
```

```
import java.io.File;


@Slf4j
@RestController
@RequestMapping("/api/ekg")
@CrossOrigin("http://localhost:4200")
public class ResultController {


    @PostMapping
    public ResponseEntity<?>
processEKG(@RequestParam("file") MultipartFile file){


        return ResponseEntity.ok().build();
    }
}


package com.eck_analytics.Controller;


import com.eck_analytics.Model.Person;
import com.eck_analytics.Services.PersonService;
import com.eck_analytics.dto.request.EditProfile;
import com.eck_analytics.security.UserPrincipal;
import
org.springframework.beans.factory.annotation.Autowired;
import
org.springframework.boot.actuate.trace.http.HttpTrace;
import org.springframework.http.ResponseEntity;
import
org.springframework.security.core.context.SecurityContextH
older;
import org.springframework.web.bind.annotation.*;


@RestController
@RequestMapping("/api/user")
@CrossOrigin("http://localhost:4200")
public class UserController{


    @Autowired
    private PersonService personService;


    @PostMapping("/edit")
    public ResponseEntity<?>
changeUserDetails(@RequestBody EditProfile editProfile){
        Person person = new
Person(editProfile.getAge(),editProfile.getWeight(),
                editProfile.getHeight(),editProfile.getSex());
        personService.savePerson(person);
        return ResponseEntity.ok().build();
    }


    @GetMapping("/info")
    public ResponseEntity<?> getUserSummary(){
        Object potentialPrincipal =
SecurityContextHolder.getContext().getAuthentication().getP
rincipal();


        if (potentialPrincipal instanceof UserPrincipal) {
            return
ResponseEntity.ok(personService.getUserSummary((((UserP
rincipal) potentialPrincipal).getId())));
        }
        return ResponseEntity.ok().build();
    }
}


package com.eck_analytics.DAO.impl;



import com.eck_analytics.DAO.AnomalyDAO;
import
com.eck_analytics.DAO.HibernateSessionFactoryUtil;
import com.eck_analytics.Model.Anomaly;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;


import java.util.List;
```

```
@Repository
public class AnomalyDAOImpl implements AnomalyDAO
{


    @Autowired
    private SessionFactory sessionFactory;


    @Override
    public Anomaly findById(int id) {
        return
sessionFactory.getCurrentSession().get(Anomaly.class, id);
    }


    @Override
    public Integer save(Anomaly anomaly) {
        int id = 0;
        Session session = sessionFactory.getCurrentSession();
        try {

anomaly.setAnomalyString(anomaly.getAnomalyString().rep
laceAll("\u0000", ""));
            session.save(anomaly);
            id = anomaly.getId();
        } catch (HibernateException he) {
        }
        return id;
    }


    @Override
    public void update(Anomaly anomaly) {

anomaly.setAnomalyString(anomaly.getAnomalyString().rep
laceAll("\u0000", ""));
        Session session =sessionFactory.getCurrentSession();
        session.update(anomaly);
    }


    @Override
    public void delete(Anomaly anomaly) {
        Session session = sessionFactory.getCurrentSession();
        session.delete(anomaly);
    }


    @Override
    public List<Anomaly> findAll() {
        return (List<Anomaly>)
sessionFactory.getCurrentSession().createQuery("From
Anomaly ").list();
    }
}


package com.eck_analytics.DAO.impl;


import com.eck_analytics.DAO.ExampleDAO;
import
com.eck_analytics.DAO.HibernateSessionFactoryUtil;
import com.eck_analytics.Model.Example;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;


import java.util.List;


@Repository
public class ExampleDAOImpl implements ExampleDAO
{


    @Autowired
    private SessionFactory sessionFactory;


    @Override
    public Example findById(int id) {
        return
sessionFactory.getCurrentSession().get(Example.class, id);
    }
```

```
    @Override
    public void save(Example example) {
        Session session = sessionFactory.getCurrentSession();
        session.save(example);
    }

    @Override
    public void update(Example example) {
        Session session = sessionFactory.getCurrentSession();
        session.update(example);
    }

    @Override
    public void delete(Example example) {
        Session session = sessionFactory.getCurrentSession();
        session.delete(example);
    }

    @Override
    public List<Example> findAll() {
        return (List<Example>)
sessionFactory.getCurrentSession().createQuery("From
Example ").list();
    }
}
```

```
package com.eck_analytics.DAO.impl;

import com.eck_analytics.DAO.PersonDAO;
import com.eck_analytics.Model.Person;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import
org.springframework.transaction.annotation.Transactional;

import java.util.List;

@Repository
public class PersonDAOImpl implements PersonDAO {
    @Autowired

 private SessionFactory sessionFactory;

    @Override
    @Transactional
    public Person findById(int id) {
        return
sessionFactory.getCurrentSession().get(Person.class, id);
    }

    @Override
    public void save(Person person) {
        int id = 0;
        Session session = sessionFactory.getCurrentSession();
        try {
            session.save(person);
            id = person.getId();
        } catch (HibernateException he) {
        }
    }

    @Override
    public void update(Person person) {
        Session session = sessionFactory.getCurrentSession();
        session.update(person);
    }

    @Override
    public void delete(Person person) {
        Session session = sessionFactory.getCurrentSession();
        session.delete(person);
    }

    @Override
    public List<Person> findAll() {
```

```
        List<Person> results = (List<Person>)
sessionFactory.getCurrentSession().createQuery("From
result").list();

        return results;

    }



}


package com.eck_analytics.DAO.impl;


import
com.eck_analytics.DAO.HibernateSessionFactoryUtil;

import com.eck_analytics.DAO.ResultDAO;

import com.eck_analytics.Model.Result;

import org.hibernate.HibernateException;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import
org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Repository;


import java.util.List;


@Repository

public class ResultDAOImpl implements ResultDAO {


    @Autowired

    private SessionFactory sessionFactory;


    @Override

    public Result findById(int id) {

        return
sessionFactory.getCurrentSession().get(Result.class, id);

    }


    @Override

    public Integer save(Result result) {

        int id = 0;

        Session session = sessionFactory.getCurrentSession();

  try {

            session.save(result);

            id = result.getId();

        } catch (HibernateException he) {

        }


        return id;

    }


    @Override

    public void update(Result result) {


result.setResultString(result.getResultString().replaceAll("\u0
000", ""));

        Session session = sessionFactory.getCurrentSession();

        session.update(result);

    }


    @Override

    public void delete(Result result) {

        Session session = sessionFactory.getCurrentSession();

        session.delete(result);

    }


    @Override

    public List<Result> findAll() {


  List<Result> results = (List<Result>)
sessionFactory.getCurrentSession().createQuery("From
Result ").list();

        return results;

    }

}

package com.eck_analytics.DAO.impl;


import
com.eck_analytics.DAO.HibernateSessionFactoryUtil;

import com.eck_analytics.DAO.TackDAO;

import com.eck_analytics.Model.Tact;
```

```
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;


import java.util.List;


@Repository
public class TackDAOImpl implements TackDAO {



    @Autowired
    private SessionFactory sessionFactory;


    @Override
    public Tact findById(int id) {
        return
sessionFactory.getCurrentSession().get(Tact.class, id);
    }


    @Override
    public Integer save(Tact tact) {
        Session session = sessionFactory.getCurrentSession();
        session.save(tact);
        return tact.getId();
    }



@Override
    public void update(Tact tact) {
        Session session = sessionFactory.getCurrentSession();
        session.update(tact);
    }


    @Override
    public void delete(Tact tact) {
        Session session = sessionFactory.getCurrentSession();
        session.delete(tact);
    }


    @Override
    public List<Tact> findAll() {
        List<Tact> results = (List<Tact>)
sessionFactory.getCurrentSession().createQuery("From Tact
").list();
        return results;
    }
}
```

```
package com.eck_analytics.DAO.impl;



import
com.eck_analytics.DAO.HibernateSessionFactoryUtil;
import com.eck_analytics.DAO.UserDAO;
import com.eck_analytics.Model.Person;
import com.eck_analytics.Model.User;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import
org.springframework.transaction.annotation.Transactional;



import java.util.List;


@Repository
public class UserDAOImpl implements UserDAO {
    @Autowired
    private SessionFactory sessionFactory;


    @Transactional
    public User findById(int id) {
        return
sessionFactory.getCurrentSession().get(User.class, id);
```

```
    }

    public User getDetailedUserByUsernameOrEmail(String
cred) {
        List<User> user = sessionFactory.getCurrentSession().
            createQuery("From User WHERE email = :email",
User.class).setParameter("email", cred).list();
        if (user.size()==0)
            user= sessionFactory.getCurrentSession().
            createQuery("From User WHERE username =
:username", User.class).setParameter("username", cred).list();
        if(user.size()==0)
            return null;
        return user.get(0);
    }

    @Override
    public Integer save(User user) {
        Session session = sessionFactory.getCurrentSession();
        session.save(user);
        Person person = new Person();
        person.setId(user.getId());

 session.save(person);
        return user.getId();

    }

    @Override
    public void update(User user) {
        Session session = sessionFactory.getCurrentSession();
        session.update(user);
    }

    @Override
    public void delete(User user) {
        Session session = sessionFactory.getCurrentSession();
        session.delete(user);
    }

    @Override

public List<User> findAll() {

  List<User> results = (List<User>)
sessionFactory.getCurrentSession().createQuery("From
User").list();
        return results;
    }
}
```

```
package com.eck_analytics.DAO;

import com.eck_analytics.Model.Anomaly;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.Transaction;
import org.springframework.stereotype.Repository;
import java.util.List;

@Repository
public interface AnomalyDAO {
    Anomaly findById(int id);

    Integer save(Anomaly anomaly);

    void update(Anomaly anomaly);

    void delete(Anomaly anomaly);

    List<Anomaly> findAll();}

package com.eck_analytics.DAO;

import com.eck_analytics.Model.Example;

import org.hibernate.Session;
import org.hibernate.Transaction;
import org.springframework.stereotype.Repository;
```

```
import java.util.List;

@Repository
public interface ExampleDAO {
    Example findById(int id);

    void save(Example example);

    void update(Example example);

    void delete(Example example);

    List<Example> findAll();
}
```

```
package com.eck_analytics.DAO;

import com.eck_analytics.Model.*;
import lombok.extern.slf4j.Slf4j;
import org.hibernate.SessionFactory;
import
org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.cfg.Configuration;

@Slf4j

public class HibernateSessionFactoryUtil {

    private static SessionFactory sessionFactory;

    private HibernateSessionFactoryUtil() {
    }

    public static SessionFactory getSessionFactory() {
        if (sessionFactory == null) {
            try {
                Configuration configuration = new
Configuration().configure();
                configuration.addAnnotatedClass(Anomaly.class);
                configuration.addAnnotatedClass(Example.class);
                configuration.addAnnotatedClass(Person.class);
                configuration.addAnnotatedClass(Result.class);
                configuration.addAnnotatedClass(Tact.class);
                StandardServiceRegistryBuilder builder = new
StandardServiceRegistryBuilder().applySettings(configuratio
n.getProperties());
                sessionFactory =
configuration.buildSessionFactory(builder.build());

            } catch (Exception e) {
                log.debug("Error with sessionFactory!: ", e);
            }
        }
        return sessionFactory;
    }
}
```

```
package com.eck_analytics.DAO;
import com.eck_analytics.Model.Person;
import java.util.List;

public interface PersonDAO {
    Person findById(int id);

    void save(Person example);

    void update(Person example);

    void delete(Person example);

    List<Person> findAll();
}
```

```
package com.eck_analytics.DAO;

import com.eck_analytics.Model.Result;
import org.hibernate.HibernateException;
import org.hibernate.Session;
```

```
import org.hibernate.Transaction;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface ResultDAO {
    Result findById(int id);

    Integer save(Result result);

    void update(Result result);

    void delete(Result result);

    List<Result> findAll();
}

package com.eck_analytics.DAO;

import com.eck_analytics.Model.Tact;
import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.Transaction;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface TackDAO {

    //TODO rewrite with generic com.diploma.DAO
    Tact findById(int id);

    Integer save(Tact tact);

    void update(Tact tact);

    void delete(Tact tact);

    List<Tact> findAll();
}

package com.eck_analytics.DAO;
import com.eck_analytics.Model.User;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface UserDAO {
    User findById(int id);


User getDetailedUserByUsernameOrEmail(String cred);

    Integer save(User user);

    void update(User user);

    void delete(User user);

    List<User> findAll();
}

package com.eck_analytics.dto.request;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class EditProfile {
    private int height;
    private int weight;
    private int age;
    private int sex;
}
```

```
package com.eck_analytics.dto.request;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import javax.validation.constraints.NotBlank;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
public class LoginRequest {

    @NotBlank
    private String usernameOrEmail;

    @NotBlank
    private String password;
}

package com.eck_analytics.dto.request;

import lombok.Getter;
import lombok.Setter;

import javax.validation.constraints.Email;
import javax.validation.constraints.NotBlank;
import javax.validation.constraints.Size;
@Getter
@Setter


public class SignUpRequest {
    @NotBlank
    @Size(min = 3, max = 64)
    private String username;

    @NotBlank
    @Size(max = 128)
    @Email
    private String email;

    @NotBlank
    @Size(min = 6, max = 128)
    private String password;

    private String role;
}

package com.eck_analytics.dto.response;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
@AllArgsConstructor
public class ApiResponse {

    private Boolean success;

   private String message;
}

package com.eck_analytics.dto.response;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class JwtAuthenticationResponse {

    private String accessToken;
```

```
    private String tokenType = "Bearer";

    public JwtAuthenticationResponse(String accessToken) {
        this.accessToken = accessToken;
    }
}
```

```
package com.eck_analytics.dto.response;


import lombok.Data;

import java.util.ArrayList;
import java.util.List;

@Data
public class ResultAnomayResponse {

    private List<AnomalyResponse> responses = new
ArrayList<>();


    @Data
    public class AnomalyResponse{
        private List<Integer> numbers;
        private List<Character> letters;
        private String type;
        private Double probability;
    }
}
```

```
package com.eck_analytics.dto.response;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class UserSummary {
    private String username;
    private String email;
    private int weight;
    private int height;
    private int age;
    private int sex;
}
```

```
package com.eck_analytics.FileSystem;

public enum FileType {
        STATISTIC_TXT("txt"),

  EXAMPLE_CSV("csv");

    private final String fileType;

    FileType(String fileType)
    {
        this.fileType = fileType;
    }

    public String getNameOfFileType() {
        return fileType;
    }
}
```

```
package com.eck_analytics.FileSystem;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class InputFile {
    private FileType fileType;
    private String pathToFile;

    public InputFile(){}
```

```
    public InputFile(FileType fileType, String pathToFile){
        this.fileType = fileType;
        this.pathToFile = pathToFile;
    }

}
```

```
package com.eck_analytics.FileSystem;

import com.eck_analytics.Model.AnomalyType;
import com.eck_analytics.Model.Example;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.regex.Pattern;
public class SimpleFileReader extends java.io.FileReader
{

    private static final Pattern CLEAR_PATTERN =
Pattern.compile("[\\s]+");

    public SimpleFileReader(InputFile file) throws
FileNotFoundException {
        super(file.getPathToFile());
    }

    /***
     *return not a lot of example that can be stored in program
memory
     * @param file - only txt file
     * @return all examples from this file
     */
    public static List<Example> readFile(InputFile file) {
        List<Example> examples = new ArrayList<>();
        if (getFileExtension(file.getPathToFile()).equals("txt"))
{
            try {

                BufferedReader fileStream = new
BufferedReader(new SimpleFileReader(file));
                fileStream.readLine();
                String nextLineInFile = fileStream.readLine();

                try {
                    while (nextLineInFile != null) {

examples.add(constructExampleFromStringTxt(nextLineInF
le));
                        nextLineInFile = fileStream.readLine();
                    }
                    fileStream.close();

                } catch (FileNotFoundException e) {
                    System.out.println("No file was read");
                } catch (IOException e) {
                    System.out.println("There was a problem reading
the file");
                }
            } catch (
                    IOException ex) {
                ex.printStackTrace();
            }

        }
        return examples;
    }
    /***
     * function for parsing string according to the pattern
     * time   id   typeOfAnomaly sub   chan   num
     * @param line - one line from txt file
     * @return new example with values from txt file
     */
    private static Example
constructExampleFromStringTxt(String line) {

        CLEAR_PATTERN.matcher(line).replaceAll("
").trim();
```

```
        String[] informationAboutExample1 = line.split(" ");
        ArrayList<String> informationAboutExample = new
ArrayList<>();
        for (String s : informationAboutExample1
        ) {
            if (!s.equals(""))
                informationAboutExample.add(s);
        }

        //time mm:ss.SSS
        String[] time =
informationAboutExample.get(0).split(":");
        int minute = Integer.parseInt(time[0]);
        char[] seconds = time[1].toCharArray();
        String secondsStr = "";
        String millisecondsStr = "";
        secondsStr += seconds[0];
        secondsStr += seconds[1];
        millisecondsStr += seconds[3];
        millisecondsStr += seconds[4];
        millisecondsStr += seconds[5];
        String numStr;
        int second = Integer.parseInt(secondsStr);
        int millisecond = Integer.parseInt(millisecondsStr);

      int millisecondSum = minute * 60000 + second * 1000 +
millisecond;

        // get all information from string in correctWay
        int previousId =
Integer.parseInt(informationAboutExample.get(1));
        String typeOfAnomaly =
informationAboutExample.get(2);
        int anomalyTime =
AnomalyType.getTypeId(typeOfAnomaly);
        int sub =
Integer.parseInt(informationAboutExample.get(3));
        int chan =
Integer.parseInt(informationAboutExample.get(4));

        if (informationAboutExample.get(5).contains("\t")) {
            numStr =
informationAboutExample.get(5).split("\t")[0];
        } else numStr = informationAboutExample.get(5);
        int num = Integer.parseInt(numStr);

        return new Example(millisecondSum, anomalyTime,
sub, chan, num, previousId);
    }

    /***
     * function for parsing string according to the pattern
     * id   first_val=mlii second_val =v5
     * @param line -one line from csv file
     * @return new example with values from cvn file
     */
    public static Example
constructExampleFromStringCsv(String line) {
        String[] informationAboutExample = line.split(",");
        int previousId =
Integer.parseInt(informationAboutExample[0]);
        int mlii =
Integer.parseInt(informationAboutExample[1]);
        int v5 = Integer.parseInt(informationAboutExample[2]);
        return new Example(previousId, mlii, v5);
    }

    public static String getFileExtension(String fullName) {
        String fileName = new File(fullName).getName();
        int dotIndex = fileName.lastIndexOf('.');
        return (dotIndex == -1) ? "" :
fileName.substring(dotIndex + 1);
    }
}

package com.eck_analytics.Model;

import lombok.Data;

import javax.persistence.*;

@Data
@Entity
```

```
@Table(name = "anomaly")

public class Anomaly {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(name = "anomaly_string")
    private String anomalyString;

    @Column(name = "anomaly_type_id")
    private int anomalyType;

    public Anomaly() {
    }

    public Anomaly(String anomalyString, int anomalyType) {
        this.anomalyString = anomalyString;
        this.anomalyType = anomalyType;
    }
}
```

```
package com.eck_analytics.Model;


public enum AnomalyType {

    NOO(0,""),
    NOT_IDENTIFIED(1,"+"),
    NORMAL(2,"N"),
    ARRHYTHMIA(3,"A"),
    VIBRATION(4,"~"),
    STOP(5,"|"),
    V(6,"V");

    private final int typeId;
    private final String typeString;

    AnomalyType(int typeId, String typeString) {
        this.typeId = typeId;
        this.typeString = typeString;
    }

    public static int getTypeId(String typeString) {
        switch (typeString){
            case("+"): return 1;
            case("N"): return 2;
            case("A"): return 3;
            case("~"): return 4;
            case("|"): return 5;
            case("V"): return 6;
        }
        return 0;
    }
}
```

```
package com.eck_analytics.Model;
import lombok.Data;
import javax.persistence.*;


@Data
@Entity
@Table(name = "example")
public class Example {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

  //special values
    @Column(name = "mlii")
    private int mlii; //first row from csv\

    @Column(name = "v5")
    private int v5; //second row from csv

    @Column(name = "example_time")
    private int timeOfExample;
```

```
    @Column(name = "type")
    private int type;

    @Column(name = "sub")
    private int sub;

    @Column(name = "chan")
    private int chan;

    @Column(name = "num")
    private int num;

    @Column(name = "letter")

    private char letter;

    //@Column(name = "result_id")
    //private int resultId ;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "result_id")
    private Result result;

    private int previous_id;



 public Example() {

    }

    public Example(int time, int type, int sub, int chan, int
num, int previous_id) {
        timeOfExample = time;
        this.type = type;
        this.sub = sub;
        this.chan = chan;
        this.previous_id = previous_id;
    }

    public Example(int previous_id, int mlii, int v5) {
        this.previous_id = previous_id;
        this.mlii = mlii;
        this.v5 = v5;
    }
}
```

```
package com.eck_analytics.Model;

import lombok.Data;

import javax.persistence.*;

@Data
@Entity
@Table(name = "person")
public class Person {
    @Id
//    @GeneratedValue(strategy =
GenerationType.IDENTITY)
    private int id;
    private int age;
    private int weight;
    private int height;
    private int sex;

    public Person(){
    }

    public Person(int age, int weight, int height, int sex) {
        this.age = age;
        this.weight = weight;
        this.height = height;
        this.sex = sex;
    }
}
```

```
package com.eck_analytics.Model;


import lombok.Data;
import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;

@Data
@Entity
@Table(name = "result")
public class Result {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(name = "result_string")
    private String resultString;

    private int anomaly;


    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "person_id")
    private Person person;

    public Result() {

   }

    public Result(String resultString,int anomaly){
        this.resultString = resultString;
        this.anomaly = anomaly;
    }
}

package com.eck_analytics.Model;

import lombok.Data;

import javax.persistence.*;
import java.sql.Timestamp;

@Data
@Entity
@Table(name = "tact")

public class Tact {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(name = "start_time")
    private Timestamp startTime;

    @Column(name = "end_time")
    private Integer endTime;

    @Column(name = "tact_string")
    private Integer tactString;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "result_id")
    private Result result;

}

package com.eck_analytics.Model;

import lombok.Data;
import lombok.Getter;
import lombok.Setter;

import javax.persistence.*;

@Getter
```

```
@Setter
@Data
@Entity
@Table(name = "user_data")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    @Column(name = "username")
    private String username;
    @Column(name = "email")
    private String email;
    @Column(name = "password")
    private String password;
    @Column(name = "is_mail_verified")
    private boolean isMailVerified;


    public User(int authorityId, String username, String
password,  String email,boolean isActive) {
        this.email = email;
        this.username = username;
        this.password = password;
        this.isMailVerified = isActive;}


    public User() {
    }
}
```

```
package com.eck_analytics.Services.impl;


import com.eck_analytics.Model.Anomaly;
import com.eck_analytics.Services.AnomalySearcher;
import com.eck_analytics.Services.AnomalyService;
import com.eck_analytics.Services.ResultService;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;


import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;


@Service
public class AnomalySearcherImpl implements
AnomalySearcher {
    private AnomalyService anomalyService;
    private ResultService resultService;


    @Autowired
    public AnomalySearcherImpl(AnomalyService
anomalyService, ResultService resultService) {
        this.anomalyService = anomalyService;
        this.resultService = resultService;
    }




    @Override
    public void getAnomalyInResult(int idOfResult) {
        List<Anomaly> anomalies =
anomalyService.findAllAnomaly();
        String currResultString =
resultService.findResult(idOfResult).getResultString();
        getAnomalyInString(currResultString, anomalies);
    }


    /***
     * check every symbol in result without changing elements
- the simplest algorithm
     * @param result -string that should be checked
     * @param anomalies -list of anomaly that can be in
linguistic chain
     * @return map where int - is start index of firs letter from
anomaly and anomaly is info about anomaly
     */
    @Override
    public Map<Integer, Anomaly>
getAnomalyInString(String result, List<Anomaly>
anomalies) {
        Map<Integer, Anomaly> anomalyMap = new
LinkedHashMap<>();
```

```
        for (Anomaly anomaly : anomalies) {
            for (int i = 0; i < result.length() -
LinguisticChainServiceImpl.CHAR_IN_ANOMALY; i++) {
                String currPartFromResult = result.substring(i, i +
LinguisticChainServiceImpl.CHAR_IN_ANOMALY);
                if (currPartFromResult.equals(anomaly)) // TODO
Check String to Object equals
                    anomalyMap.put(i, anomaly);
            }
        }
        return anomalyMap;
    }
}
```

```
package com.eck_analytics.Services.impl;

import com.eck_analytics.DAO.AnomalyDAO;
import com.eck_analytics.Model.Anomaly;
import com.eck_analytics.Services.AnomalyService;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class AnomalyServiceImpl implements
AnomalyService {
    //TODO generic service
    private AnomalyDAO anomalyDAO;

    @Autowired
    public AnomalyServiceImpl(AnomalyDAO anomalyDAO)
{
        this.anomalyDAO = anomalyDAO;
    }

    @Override
    public Anomaly findAnomaly(int id) {
        return anomalyDAO.findById(id);
    }

    @Override
    public void saveAnomaly(Anomaly anomaly) {
        anomalyDAO.save(anomaly);
    }

    @Override
    public void deleteAnomaly(Anomaly anomaly) {
        anomalyDAO.delete(anomaly);
    }

    @Override
    public void updateAnomaly(Anomaly anomaly) {
        anomalyDAO.update(anomaly);
    }

    @Override
    public List<Anomaly> findAllAnomaly() {
        return anomalyDAO.findAll();
    }
}
```

```
package com.eck_analytics.Services.impl;

import com.eck_analytics.DAO.UserDAO;
import com.eck_analytics.Model.User;
import com.eck_analytics.Utils.Constants;
import com.eck_analytics.dto.request.LoginRequest;
import com.eck_analytics.dto.request.SignUpRequest;
import lombok.extern.slf4j.Slf4j;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.security.authentication.AuthenticationManager;

import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;

import org.springframework.security.core.Authentication;

import org.springframework.security.core.context.SecurityContextHolder;


import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.eck_analytics.security.JwtTokenProvider;


@Slf4j

@Service

@Transactional

public class AuthService {


   @Autowired

   private UserDAO userDao;




   @Transactional

   public ResponseEntity registerUser(SignUpRequest request, PasswordEncoder encoder) {


      if (userDao.getDetailedUserByUsernameOrEmail(request.getUsername())!=null) {

         return Constants.ResponseEntities.BAD_REQ_USERNAME_TAKEN;

      }

      if (userDao.getDetailedUserByUsernameOrEmail(request.getEmail())!=null) {

         return Constants.ResponseEntities.BAD_REQ_EMAIL_TAKEN;

      }

      //change authority

      User user = new User(1, request.getUsername(),

            encoder.encode(request.getPassword()), request.getEmail(),false);



      userDao.save(user);



      return Constants.ResponseEntities.USER_REGISTERED_SUCCESSFULLY;

   }


   public String authenticateUser(LoginRequest request, AuthenticationManager authManager, JwtTokenProvider tokenProvider) {

      log.debug("Request: {}|{}", request.getUsernameOrEmail(), request.getPassword());

      Authentication authentication = authManager.authenticate(

            new UsernamePasswordAuthenticationToken(

                  request.getUsernameOrEmail(),

                  request.getPassword()

            )

      );



SecurityContextHolder.getContext().setAuthentication(authentication);


      return tokenProvider.generateToken(authentication);

   }

}


package com.eck_analytics.Services.impl;



import com.eck_analytics.Algorithm.DamerauLevenshtein;

import com.eck_analytics.Algorithm.XCompression;

import com.eck_analytics.Algorithm.YCompression;

import com.eck_analytics.Model.Anomaly;

import com.eck_analytics.Services.AnomalySearcher;

import com.eck_analytics.Services.AnomalyService;

import com.eck_analytics.Services.EKGAnalyze;
```

```
import com.eck_analytics.Services.LinguisticChainService;

import com.eck_analytics.Utils.Alphabet;

import com.eck_analytics.Utils.Constants;

import com.eck_analytics.Utils.LinguisticChainBuilder;


import
com.eck_analytics.dto.response.ResultAnomayResponse;

import
org.apache.commons.text.similarity.HammingDistance;

import
org.apache.commons.text.similarity.JaroWinklerDistance;

import
org.apache.commons.text.similarity.LevenshteinDistance;

import
org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.multipart.MultipartFile;


import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.util.List;



public class EKGAnalyzeImpl implements EKGAnalyze {

    LevenshteinDistance levenshtein = new
LevenshteinDistance();

    JaroWinklerDistance jaroWinkler = new
JaroWinklerDistance();

    HammingDistance hammingDistance = new
HammingDistance();

    DamerauLevenshtein damerauLevenshtein = new
DamerauLevenshtein();

    YCompression yCompression = new YCompression();

    XCompression xCompression = new XCompression();


    @Autowired

    private AnomalyService anomalyService;


    @Override

    public ResultAnomayResponse
getEKGresult(MultipartFile file) throws IOException {

        String currString = "";

        ResultAnomayResponse resultAnomayResponse = new
ResultAnomayResponse();


        BufferedReader br;

        try {


            String line;

            InputStream is = file.getInputStream();

            br = new BufferedReader(new
InputStreamReader(is));


while ((line = br.readLine()) != null) {

                //LINE PROCESSING

                Integer value = Integer.valueOf(line);

                if (currString.length() <
Constants.LinguisticConstant.ANOMALYSIZE) {

                    currString = currString +
LinguisticChainBuilder.getLetter(value,
Alphabet.TEST_ARRAY);

                } else {



                    int length = currString.length();

                    currString = currString.substring(0, length - 1);

                    currString = currString +
LinguisticChainBuilder.getLetter(value,
Alphabet.TEST_ARRAY);

                }



            }


        } catch (IOException e) {

            System.err.println(e.getMessage());

        }

        return resultAnomayResponse;

    }


    public double probabilityOfAnomaly(String curr) {

        double maxComparison =0 ;
```

```
List<Anomaly> allAnomaly =
anomalyService.findAllAnomaly();

        for (Anomaly anomaly : allAnomaly

        ) {

            if (anomaly.getAnomalyString().compareTo(curr) ==
0)

                return 1.0;



    }

        //   Y  COMPARISON

        for (Anomaly anomaly : allAnomaly

        ) {

            char[] releaseAnomaly =
yCompression.release(anomaly.getAnomalyString().toCharA
rray(), 1.2);

            if (releaseAnomaly.toString().compareTo(curr) == 0)

                return 1.0;

            releaseAnomaly =
yCompression.release(anomaly.getAnomalyString().toCharA
rray(), 0.8);

            if (releaseAnomaly.toString().compareTo(curr) == 0)

                return 1.0;

        }

        //   X  COMPARISON

        for (Anomaly anomaly : allAnomaly

        ) {

            String anomalyString = "";

            double border =
anomaly.getAnomalyString().length() * 0.1;

            char[] releaseAnomaly =
xCompression.release(anomaly.getAnomalyString().toCharA
rray(), 1.2);

            for (int i = ((int) border); i < releaseAnomaly.length -
border; i++)

                anomalyString = anomalyString +
releaseAnomaly[i];

            if (anomalyString.compareTo(curr) == 0)

                return 1.0;



            border = curr.length() * 0.1;

            anomalyString = "";

            char[] currStr = curr.toCharArray();

            releaseAnomaly =
xCompression.release(anomaly.getAnomalyString().toCharA
rray(), 0.8);

            for (int i = ((int) border); i < curr.length() - border;
i++)

                anomalyString = anomalyString + currStr[i];

            if
(releaseAnomaly.toString().compareTo(anomalyString) == 0)



        return 1.0;

        }

        for (Anomaly anomaly : allAnomaly

        ) {

            double compareWithDistance =
compareWithDistance(anomaly.getAnomalyString(), curr);

            if (compareWithDistance>maxComparison){

                maxComparison = compareWithDistance;

            }

        }



        return maxComparison;

    }



    private double compareWithDistance(String curr, String
anomaly) {

        Double d1 = jaroWinkler.apply(curr, anomaly);

        Double d2 = 1 - ((double) levenshtein.apply(curr,
anomaly) / curr.length());

        Double d3 = 1 - ((double) hammingDistance.apply(curr
anomaly) / curr.length());

        Double d4 = 1 - ((double)
damerauLevenshtein.calculateDistance(curr, anomaly) /
curr.length());



        return (d1 + d2 + d3 + d4) / 4;

    }

}
```

```
package com.eck_analytics.Services.impl;

import com.eck_analytics.DAO.ExampleDAO;

import com.eck_analytics.Model.Example;

import com.eck_analytics.Services.ExampleService;

import
org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.context.annotation.Lazy;
import org.springframework.stereotype.Service;

import java.util.List;

@Lazy
@Service
public class ExampleServiceImpl implements
ExampleService {
    private ExampleDAO exampleDAO;

    @Autowired
    public ExampleServiceImpl(ExampleDAO exampleDAO)
{
        this.exampleDAO = exampleDAO;
    }

    @Override
    public Example findExample(int id) {
        return exampleDAO.findById(id);
    }

    @Override
    public void saveExample(Example example) {
        exampleDAO.save(example);
    }

    @Override

    public void deleteExample(Example example) {
        exampleDAO.delete(example);
    }

    @Override
    public void updateExample(Example example) {
        exampleDAO.update(example);
    }

    @Override
    public List<Example> findAllExample() {
        return exampleDAO.findAll();
    }

    @Override
    public void saveExamples(List<Example> examples){
        for (Example example:examples
            ) {
            exampleDAO.save(example);
        }
    }
}
```

```
package com.eck_analytics.Services.impl;

import com.eck_analytics.FileSystem.FileType;
import com.eck_analytics.FileSystem.InputFile;
import com.eck_analytics.FileSystem.SimpleFileReader;
import com.eck_analytics.Model.Anomaly;
import com.eck_analytics.Model.Example;
import com.eck_analytics.Model.Result;
import com.eck_analytics.Services.AnomalyService;
import com.eck_analytics.Services.ExampleService;
import com.eck_analytics.Services.LinguisticChainService;
import com.eck_analytics.Services.ResultService;
import com.eck_analytics.Utils.Alphabet;
import com.eck_analytics.Utils.Constants;
import com.eck_analytics.Utils.LinguisticChainBuilder;
//import org.apache.commons.csv.writer.CSVWriter;
import
org.junit.jupiter.params.shadow.com.univocity.parsers.csv.Cs
v;

import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Lazy;
import org.springframework.stereotype.Service;
import
org.springframework.transaction.annotation.Transactional;
import java.io.*;
import java.nio.ByteBuffer;
import java.nio.charset.StandardCharsets;
```

```
import java.util.ArrayList;

import java.util.List;


@Lazy

@Service

public class LinguisticChainServiceImpl implements
LinguisticChainService {

   public static final int CHAR_IN_ANOMALY =
Constants.LinguisticConstant.ANOMALYSIZE;


   private ExampleService exampleService;

   private ResultService resultService;

   private AnomalyService anomalyService;

   private boolean isAnomalyNow;

   private int typeOfAnomaly;


   @Autowired


public LinguisticChainServiceImpl(ExampleService
exampleService, ResultService resultService,
AnomalyService anomalyService) {

      this.exampleService = exampleService;

      this.resultService = resultService;

      this.anomalyService = anomalyService;


      this.isAnomalyNow = false;

      this.typeOfAnomaly = 0;

   }


   @Transactional

   @Override

   public void getExamplesFromFiles(String fileCSVPath,
String fileTxtPath) {


      InputFile inputCsv =
SimpleFileReader.getFileExtension(fileCSVPath).equals("cs
v") ? new InputFile(FileType.EXAMPLE_CSV,
fileCSVPath) : null;

      InputFile inputTxt =
SimpleFileReader.getFileExtension(fileTxtPath).equals("txt")
? new InputFile(FileType.STATISTIC_TXT, fileTxtPath) :
null;

      List<Example> examplesTxt =
SimpleFileReader.readFile(inputTxt);

      //save here last 10 examples and when we find anomaly
save 10 after anomaly and save them into bd

      List<Example> examplesForAnomaly = new
ArrayList<>();



      Result result = new Result("", 0);

      int resultId = resultService.saveResult(result);

      result.setId(resultId);

      try {

         BufferedReader fileStream = new
BufferedReader(new SimpleFileReader(inputCsv));

         fileStream.readLine();

         String nextLineInFile = fileStream.readLine();



  try {

            while (nextLineInFile != null) {

               Example currExample =
saveActionForExample(nextLineInFile, result, examplesTxt);

               if (currExample != null)

                  checkAnomaly(currExample,
examplesForAnomaly, result);

               nextLineInFile = fileStream.readLine();

            }

            fileStream.close();


         } catch (FileNotFoundException e) {


 System.out.println("No file was read");

         } catch (IOException e) {

            System.out.println("There was a problem reading
the file");

         }

      } catch (

            IOException ex) {

         ex.printStackTrace();

      }

   }


   public void saveChainIntoFile(String fileCSVPath, String
fileTxtPath,String fileName) throws IOException {
```

```
        InputFile inputCsv =
SimpleFileReader.getFileExtension(fileCSVPath).equals("cs
v") ? new InputFile(FileType.EXAMPLE_CSV,
fileCSVPath) : null;

        InputFile inputTxt =
SimpleFileReader.getFileExtension(fileTxtPath).equals("txt")
? new InputFile(FileType.STATISTIC_TXT, fileTxtPath) :
null;

        List<Example> examplesTxt =
SimpleFileReader.readFile(inputTxt);

        //save here last 10 examples and when we find anomaly
save 10 after anomaly and save them into bd

        List<Example> examplesForAnomaly = new
ArrayList<>();



        Result result = new Result("", 0);


        String csv = "Example_1.csv";

        FileWriter writer = new FileWriter(csv);

        File csvOutputFile = new File(fileName+".csv");

        List<String> dataLines = new ArrayList<>();

        PrintWriter pw = new PrintWriter(csvOutputFile);


        try {


            BufferedReader fileStream = new
BufferedReader(new SimpleFileReader(inputCsv));

            fileStream.readLine();

            String nextLineInFile = fileStream.readLine();


            try {

                while (nextLineInFile != null) {

                    Example currExample =
saveActionForExample(nextLineInFile, result, examplesTxt);

                    if (currExample != null)

                        {   String record = "";


record=record+(String.valueOf(currExample.getV5())+",");

                            record= record+currExample.getLetter();

                            record=record+(",");

                            if(currExample.getType()!=2){

                                record=record+("+,");

                            }

                            else record=record+("-,");


record=record+(String.valueOf(currExample.getType())+"")

                            pw.println(record);

                            writer.write(record);

                        }



                nextLineInFile = fileStream.readLine();

                }

                writer.close();

                fileStream.close();


            } catch (FileNotFoundException e) {

                System.out.println("No file was read");

            } catch (IOException e) {

                System.out.println("There was a problem reading
the file");

                System.out.println("exaption"+e.getMessage());


        }

        } catch (

                IOException ex) {

            ex.printStackTrace();

            System.out.println(ex.getMessage());

        }

        System.out.println("Operation is done");


    }


    /***

     * provide searching in list from txt file example with
similar id

     * also set all params from this txt file

     * @param examples - list of all examples from txt file

     * @param example - current example

     */

    private void getSimilarExampleByPrevId(List<Example>
examples, Example example) {

        Example exampleFromTxt = null;

        for (Example e : examples
```

```
        ) {


 if (e.getPrevious_id() == example.getPrevious_id())
                exampleFromTxt = e;
        }


        if (exampleFromTxt != null) {
            example.setChan(exampleFromTxt.getChan());
            example.setNum(exampleFromTxt.getNum());
            example.setSub(exampleFromTxt.getSub());

example.setTimeOfExample(exampleFromTxt.getTimeOfEx
ample());
            example.setType(exampleFromTxt.getType());
        }


else {
            example.setChan(0);
            example.setNum(0);
            example.setSub(0);
            example.setTimeOfExample(0);
            example.setType(0);
        }
    }


    private Example saveActionForExample(String
nextLineInFile, Result result, List<Example> examplesTxt) {
        Example currExample =
SimpleFileReader.constructExampleFromStringCsv(nextLine
InFile);
        getSimilarExampleByPrevId(examplesTxt,
currExample);
        currExample.setResult(result);
        if
(LinguisticChainBuilder.getLetter(currExample.getV5(),
Alphabet.TEST_ARRAY) != ' ') {

currExample.setLetter(LinguisticChainBuilder.getLetter(curr
Example.getV5(), Alphabet.TEST_ARRAY));
            result.setResultString(result.getResultString() +
currExample.getLetter());
            //exampleService.saveExample(currExample);


    resultService.updateResult(result);
        } else return null;
        return currExample;
    }


    private void checkAnomaly(Example currExample,
List<Example> examplesForAnomaly, Result result) {
        //TODO Move
examplesForAnomaly.add(currExample); here as it is presen
in every branch of this method
        if (currExample.getType() > 2) {


            isAnomalyNow = true;
            typeOfAnomaly = currExample.getType();


  examplesForAnomaly.add(currExample);
            result.setAnomaly(result.getAnomaly() + 1);
            resultService.updateResult(result);
        } else {
            if (!isAnomalyNow) {
                if (examplesForAnomaly.size() >=
CHAR_IN_ANOMALY / 2) {
                    examplesForAnomaly.remove(0);
                    examplesForAnomaly.add(currExample);
                } else examplesForAnomaly.add(currExample);
            } else {
                if (examplesForAnomaly.size() >=
CHAR_IN_ANOMALY - 1) {
                    examplesForAnomaly.add(currExample);
                    String anomaly = new String();
                    for (Example e : examplesForAnomaly) {
                        anomaly = anomaly+(e.getLetter());
                    }



  anomalyService.saveAnomaly(new Anomaly(anomaly,
typeOfAnomaly));
                    isAnomalyNow = false;
                    typeOfAnomaly = 0;
                } else examplesForAnomaly.add(currExample);
            }
        }
```

```
    }
}

package com.eck_analytics.Services.impl;

import com.eck_analytics.DAO.PersonDAO;
import com.eck_analytics.DAO.UserDAO;
import com.eck_analytics.Model.Person;

import com.eck_analytics.Model.User;
import com.eck_analytics.Services.PersonService;
import com.eck_analytics.dto.response.UserSummary;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class PersonServiceImpl implements PersonService
{
    private PersonDAO personDAO;
    private UserDAO userDAO;

    @Autowired
    public PersonServiceImpl(PersonDAO personDAO,
UserDAO userDAO) {
        this.personDAO = personDAO;
        this.userDAO = userDAO;
    }

    @Override
    public Person getById(int id) {
        return personDAO.findById(id);
    }

    @Override
    public void savePerson(Person person) {
        personDAO.save(person);
    }

    @Override

    public void deletePerson(Person person) {
        personDAO.delete(person);
    }

    @Override
    public void updatePerson(Person person) {
        personDAO.update(person);
    }

    @Override
    public UserSummary getUserSummary(int id) {
        UserSummary userSummary = new UserSummary();
        User user = userDAO.findById(id);
        Person person = personDAO.findById(id);
        userSummary.setUsername(user.getUsername());
        userSummary.setEmail(user.getEmail());

        if (person != null) {
            userSummary.setAge(person.getAge());
            userSummary.setHeight(person.getHeight());
            userSummary.setWeight(person.getWeight());
            userSummary.setSex(person.getSex());
        }

        return userSummary;
    }
}

package com.eck_analytics.Services.impl;

import com.eck_analytics.DAO.ResultDAO;

import com.eck_analytics.Model.Result;
import com.eck_analytics.Services.ResultService;
```

```
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.List;

@Service
public class ResultServiceImpl implements ResultService
{
   private ResultDAO resultDAO;

   @Autowired
   public ResultServiceImpl(ResultDAO resultDAO) {
      this.resultDAO = resultDAO;
   }

   @Override
   public Result findResult(int id) {
      return resultDAO.findById(id);
   }

   @Override
   public int saveResult(Result result) {
     return resultDAO.save(result);
   }

   @Override
   public void deleteResult(Result result) {
      resultDAO.delete(result);
   }

   @Override
   public void updateResult(Result result) {
      resultDAO.update(result);
   }
   @Override
   public List<Result> findAllResult() {
      return resultDAO.findAll();
   }
}
```

```
package com.eck_analytics.Services.impl;

import com.eck_analytics.DAO.TackDAO;
import com.eck_analytics.Model.Tact;

import com.eck_analytics.Services.TackService;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.List;

@Service
public class TackServiceImpl implements TackService {
   private TackDAO tackDAO;

   @Autowired
   public TackServiceImpl(TackDAO tackDAO) {
      this.tackDAO = tackDAO;
   }
   @Override
   public Tact findAnomaly(int id) {
      return tackDAO.findById(id);
   }

   @Override
   public void saveAnomaly(Tact tact) {
      tackDAO.save(tact);
   }

   @Override
   public void deleteAnomaly(Tact tact) {
      tackDAO.delete(tact);
   }

   @Override
```

```
    public void updateAnomaly(Tact tact) {
        tackDAO.update(tact); }

    @Override
    public List<Tact> findAllAnomaly() {
        return tackDAO.findAll();
    }
}
```

```
package com.eck_analytics.Services;

import com.eck_analytics.Model.Anomaly;
import org.springframework.stereotype.Service;

import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;

@Service
public interface AnomalySearcher {
    void getAnomalyInResult(int idOfResult);
    /***
     * check every symbol in result without changing elements - the simplest algorithm
     * @param result -string that should be checked
     * @param anomalies -list of anomaly that can be in linguistic chain
     * @return map where int - is start index of firs letter from anomaly and anomaly is info about anomaly
     */
    Map<Integer, Anomaly> getAnomalyInString(String result, List<Anomaly> anomalies);
}
```

```
package com.eck_analytics.Services;

import com.eck_analytics.DAO.AnomalyDAO;
import com.eck_analytics.Model.Anomaly;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public interface AnomalyService {
    Anomaly findAnomaly(int id);

    void saveAnomaly(Anomaly anomaly);

    void deleteAnomaly(Anomaly anomaly);

    void updateAnomaly(Anomaly anomaly);

    List<Anomaly> findAllAnomaly();
}
```

```
package com.eck_analytics.Services;

import com.eck_analytics.dto.response.ResultAnomayResponse;
import org.springframework.web.multipart.MultipartFile;
import java.io.IOException;
public interface EKGAnalyze {
    ResultAnomayResponse getEKGresult(MultipartFile file) throws IOException;
}
```

```
package com.eck_analytics.Services;

import com.eck_analytics.DAO.ExampleDAO;
import com.eck_analytics.Model.Example;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public interface ExampleService {
    Example findExample(int id);

    void saveExample(Example example);

    void deleteExample(Example example);
```

```
    void updateExample(Example example);

    List<Example> findAllExample();

    void saveExamples(List<Example> examples);
}
```

```
package com.eck_analytics.Services;

import com.eck_analytics.FileSystem.FileType;
import com.eck_analytics.FileSystem.InputFile;
import com.eck_analytics.FileSystem.SimpleFileReader;
import com.eck_analytics.Model.Anomaly;
import com.eck_analytics.Model.Example;
import com.eck_analytics.Model.Result;

import com.eck_analytics.Utils.Alphabet;
import com.eck_analytics.Utils.LinguisticChainBuilder;
import org.springframework.stereotype.Service;
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

@Service
public interface LinguisticChainService {
    void getExamplesFromFiles(String fileCSVPath, String
fileTxtPath);
    void saveChainIntoFile(String fileCSVPath, String
fileTxtPath,String fileName) throws IOException;
}
```

```
package com.eck_analytics.Services;
import com.eck_analytics.Model.Person;
import com.eck_analytics.dto.response.UserSummary;
import org.springframework.stereotype.Service;

@Service
public interface PersonService {
    Person getById(int id);

    void savePerson(Person person);

    void deletePerson(Person person);

    void updatePerson(Person person);

    UserSummary getUserSummary(int id);
}
```

```
package com.eck_analytics.Services;
import com.eck_analytics.DAO.ResultDAO;
import com.eck_analytics.Model.Result;
import org.springframework.stereotype.Service;
import java.util.List;

@Service
public interface ResultService {
    Result findResult(int id);

    int saveResult(Result result);

    void deleteResult(Result result);

    void updateResult(Result result);

    List<Result> findAllResult();

}
```

```
package com.eck_analytics.Services;

import com.eck_analytics.DAO.TackDAO;
import com.eck_analytics.Model.Tact;
import org.springframework.stereotype.Service;
```

```
import java.util.List;

@Service
public interface TackService {
    Tact findAnomaly(int id);

    void saveAnomaly(Tact tact);

    void deleteAnomaly(Tact tact);

    void updateAnomaly(Tact tact);

    List<Tact> findAllAnomaly();
}
```

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

“ЗАТВЕРДЖЕНО”

В.о. завідувача кафедри

_____ Олександр ПАВЛОВ

“___” _____ 2020 р.

Веб-застосування для виявлення аномалій в електрокардіограмах лінгвістичним методом

Графічний матеріал

КПІ.ІП-6125.045420.05.99

“ПОГОДЖЕНО”

Керівник проєкту:

_____ Ю. О. Олійник

Нормоконтроль:

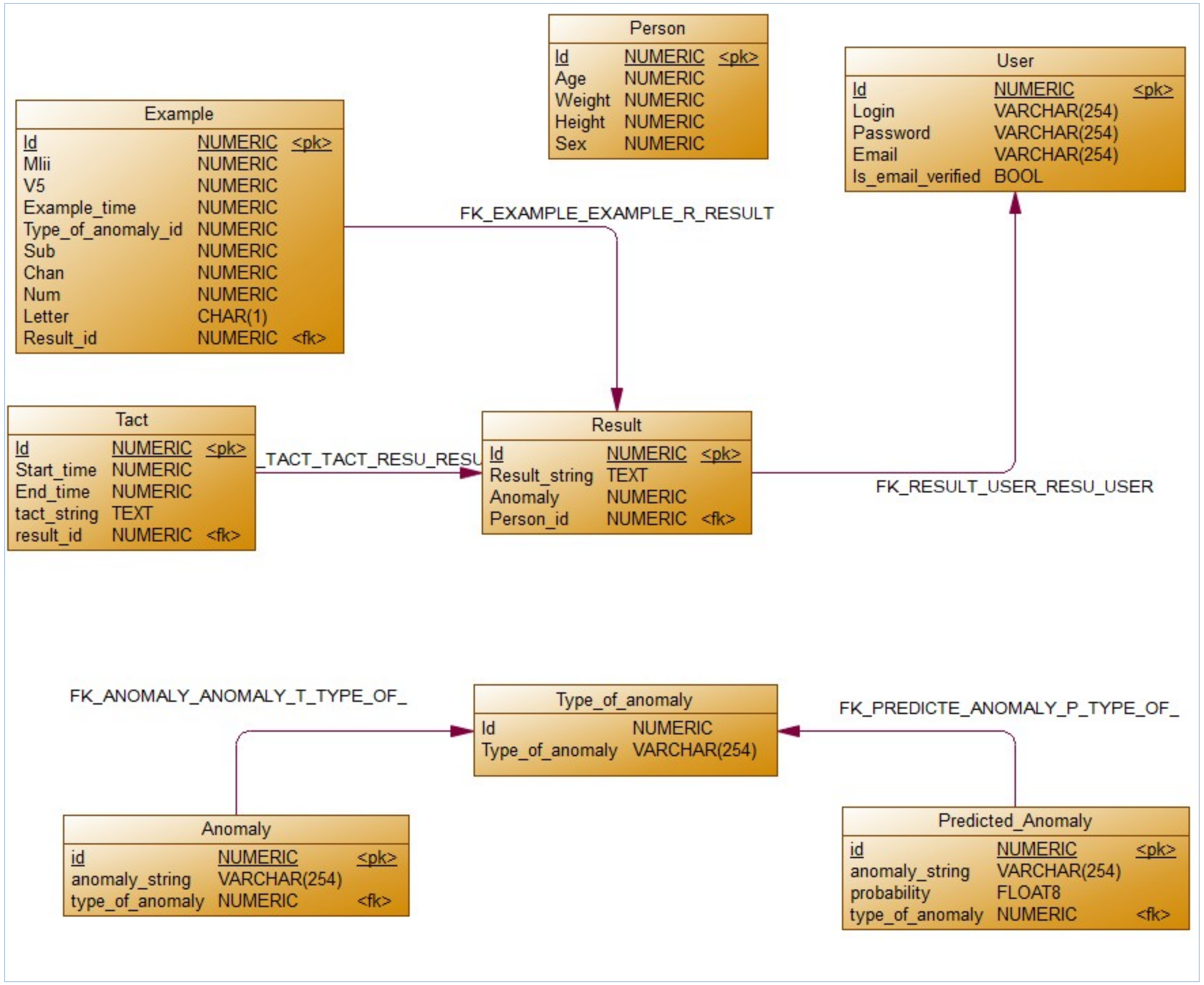
_____ К.І. Ліщук

Виконавець:

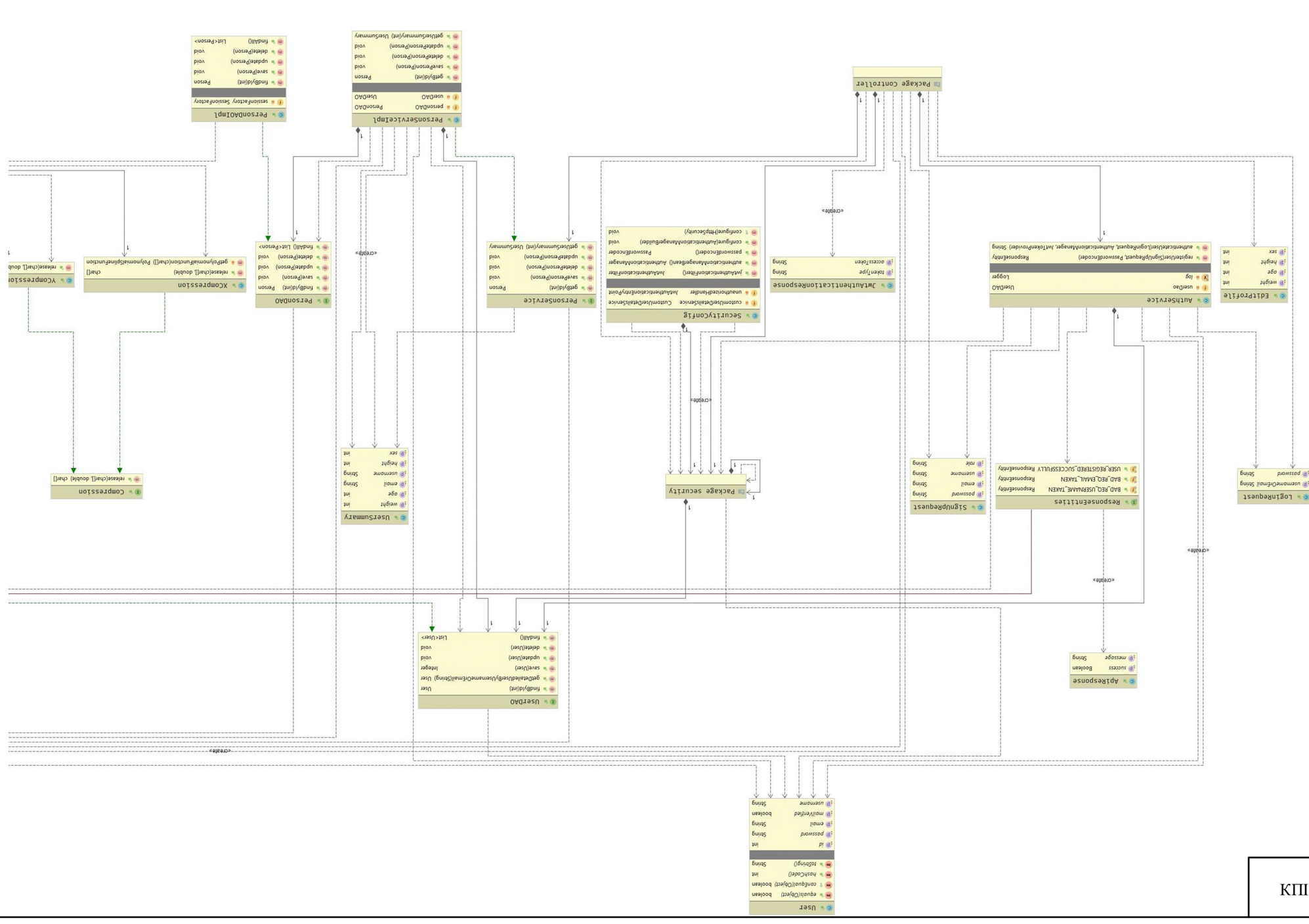
_____А.В. Цицилюк

Київ – 2020 року

КПІ.ІП-6125. 045420.05.99 СБД



Зм.	Арк.	№ документа	Підпис	Дата	КПІ.ІП-6125. 045420.05.99 СБД
Розробив		Цицилюк А.В.			Схема бази даних
Перевірив		Олійник Ю.О.			Літера / Маса / Масштаб
Т. кон.					Аркуш / Аркушів
Н. кон.		Ліщук К.І.			Веб-додаток для виявлення аномалій в електрокардіогрмах лінгвістичним методом
Затвердив		Олійник Ю.О.			КПІ ім.Ігоря Сікорського Кафедра АСОІУ гр. ІП-61

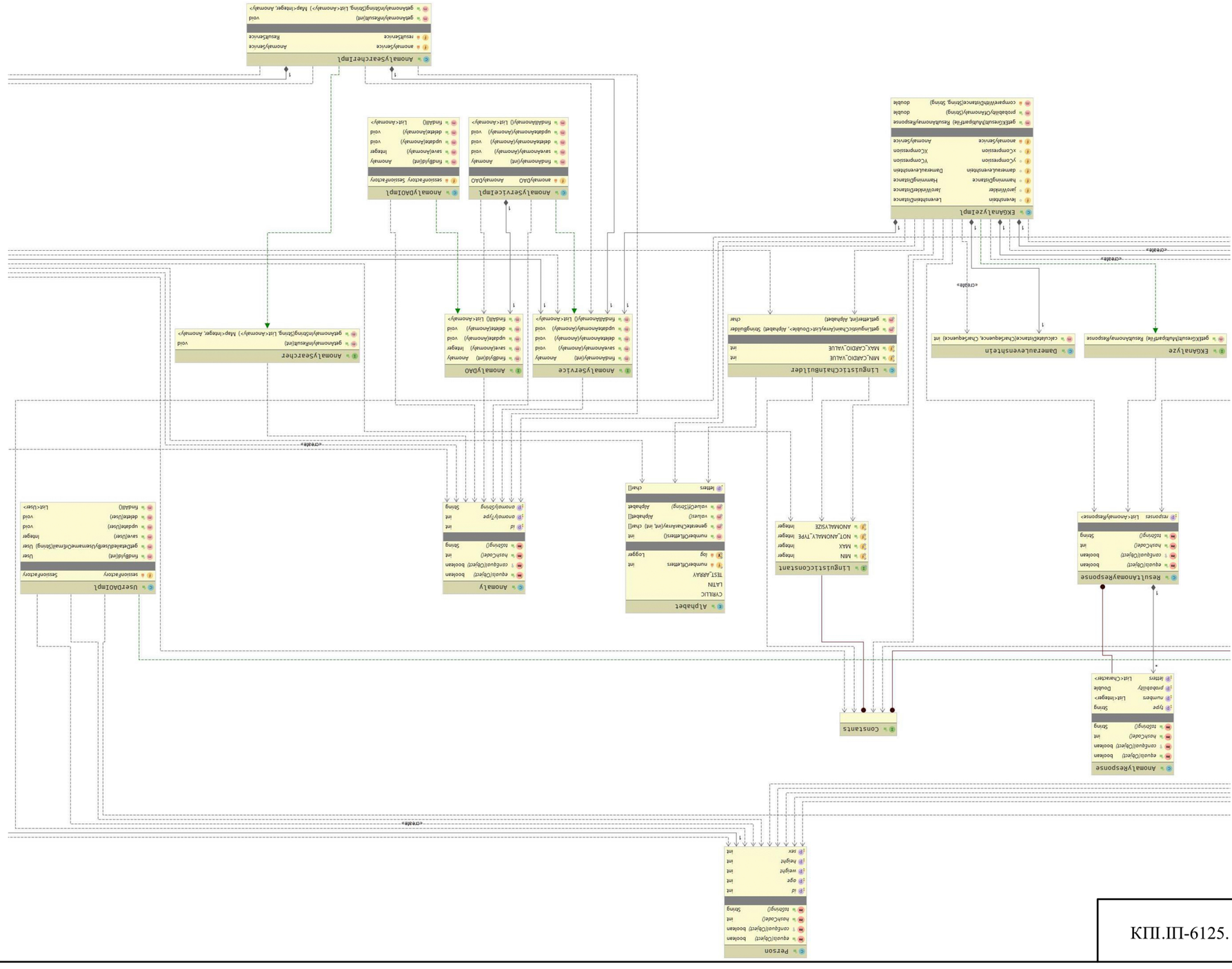


Изм.	Арк.	№ докумекта	Підпис	Дата	КПП.ПП-6125. 045420.05.99 СС
Розробив		Цицилюк А.В.			
Перевірив		Оліфнік Ю.О.			Схема структура класів програмного забезпечення
Т. кон.					
Н. кон.		Плшук К.І.			Веб-додаток для виявлення аномалій в електрокардіограмах дистанційним методом
Затвердив		Оліфнік Ю.О.			

Літера	Маса	Масштаб

Аркуш 1	Аркушів 3

КПІ ім.Ігоря Сікорського
Кафедра АСОІУ
гр. ІП-61



КПИ.ІП-6125. 045420.05.99 СС

Зм.	Арк.	№ документа	Підпис	Дата	КПИ.ІП-6125. 045420.05.99 СС		
Розробив		Цицилюк А.В.			Схема структура класів програмного забезпечення	Літера	Маса	Масштаб
Перевірив		Оніщик Ю.О.						
Т. кон.						Аркуш 2	Аркушів 3	
Н. кон.		Лішук К.І.			Веб-додаток для виявлення аномалій в електрокардіограмах дистанційним методом	КПІ ім.Ігоря Сікорського Кафедра АСОІУ гр. ІП-61		
Затвердив		Оніщик Ю.О.						

Изм.	Арк.	№ документа	Підпис	Дата
Розробив		Цицилюк А.В.		
Перевірив		Оніщик Ю.О.		
Т. контр.				
Н. контр.		Лішук К.І.		
Затвердив		Оніщик Ю.О.		

КПІ.ІП-6125. 045420.05.99 СС

Схема структура класів програмного забезпечення

Літера | Маса | Масштаб

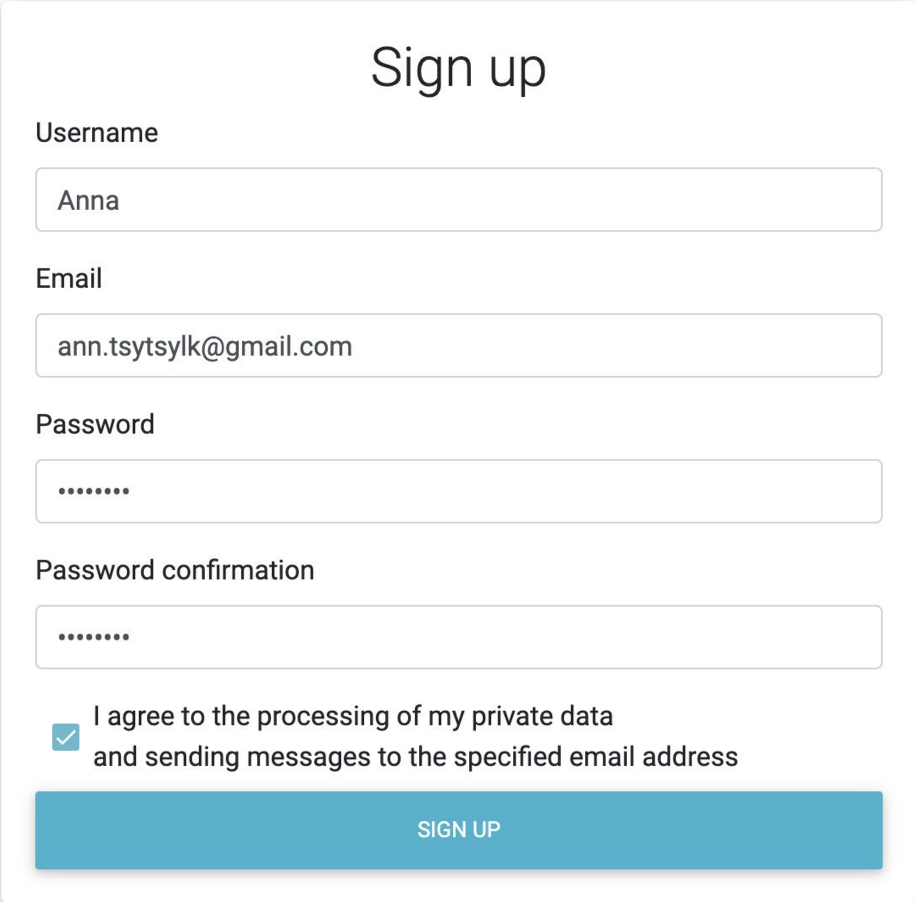
Аркуш 3 | Аркушів 3

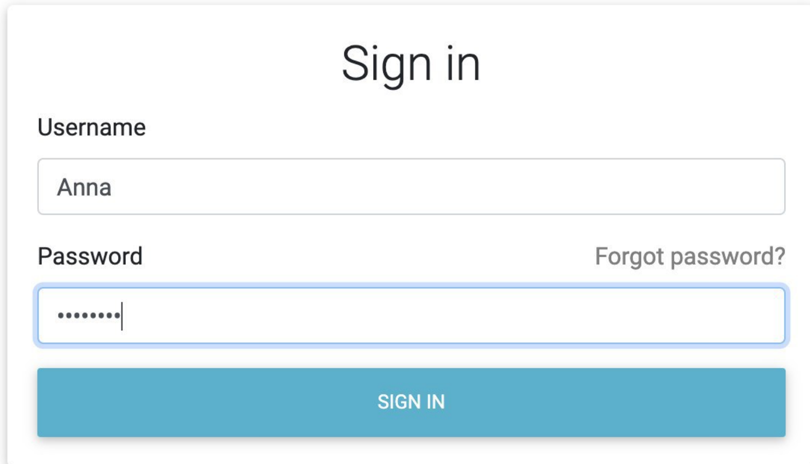
Веб-додаток для виявлення аномалій в електрокардіограмах дихательними методом

КПІ ім.Ігоря Сікорського
Кафедра АСОІУ
гр. ІП-61

КПІ.ІП-6125. 045420.05.99 СС

КПІ.ІП-6125. 045420.05.99 КЕ





					КПІ.ІП-6125. 045420.05.99 КЕ			
Зм.	Арк.	№ документа	Підпис	Дата	Креслення вигляду екранних форм	Літера	Маса	Масштаб
Розробив		Цицилюк А.В.						
Перевірив		Олійник Ю.О.				Аркуш		Аркушів
Т. кон.					Веб-додаток для виявлення аномалій в електрокардіогрмах лінгвістичним методом	КПІ ім.Ігоря Сікорського Кафедра АСОІУ гр. ІП-61		
Н. кон.		Ліщук К.І.						
Затвердив		Олійник Ю.О.						